Page 48 Item (e)
RFP Page 53 Item (e)

EASTPORT STUDY GROUP

SUMMER STUDY 1985

A REPORT TO THE
DIRECTOR
STRATEGIC DEFENSE INITIATIVE ORGANIZATION

DEPARTMENT OF DEFENSE
STRATEGIC DEFENSE INITIATIVE ORGANIZATION
WASHINGTON, DC 20301-7100

DEPARTMENT OF DEFENSE
STRATEGIC DEFENSE INITIATIVE ORGANIZATION
WASHINGTON, DC 20301-7100

14 FEB 1986

SY

MEMORANDUM FOR THE RECORD

SUBJECT: The Eastport Study Group Report

The SDIO has recently received the report of the Eastport Study Group, the panel from industry, government and academia that was appointed "to devise an appropriate computational/communications response to the (strategic defense battle management) problem and make recommendations for a research and technology development program to implement the response." A copy of this report is attached.

The essence of the report is captured by the following quote from it:

> "The panel concludes that computing resources and battle management software for strategic defense systems are within the capabilities of the hardware and software technologies that could be developed within the next several years. However, the anticipated complexity of the battle management software and necessity to test, simulate, modify, and evolve the system make battle management and command, control and communication (BM/C3) the paramount strategic defense problem."

This is in agreement with the SDIO's own assessment that the BM/C3 is potentially the "long pole in the tent."

To address this problem the report offers a research strategy that empasizes the interaction of BM/C3 with system architecture. In this vein, the panel assesses that:

> "Software technology is developing against inflexible limits in the complexity and reliability that can be achieved. The tradeoffs necessary to make the software task tractable are in the system architecture. As indicated in the Fletcher Report, the 'applique approach' of designing the system first and then writing the software to control it is the wrong approach for SDI. System Architecture and Battle Management must be developed together."

With this perspective, a number of specific observations and recommendations are made relative to technology research in computing software, processing hardware, and communications. The innovation and potential payoff of the recommended approaches are recognized. Where appropriate the SDIO will augment its BM/C3 technology research to incorporate the recommended steps, even as the SDI seeks innovation from all quarters and from all technologies.

The SDIO endorses the spirit and content of the report of the Eastport Study Group. It is found to be in harmony with the needs of the SDI program and its rapid implementation shall be pursued throughout the R&D effort.

James A. Abrahamson

JAMES A. ABRAHAMSON
Lieutenant General, USAF
Director

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT UNLIMITED DISTRIBUTION
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Eastport Study Group — 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Strategic Defense Initiative Organization
6c. ADDRESS (City, State, and ZIP Code) Marina Del Rey, California	7b. ADDRESS (City, State, and ZIP Code) SDIO/SY THE PENTAGON Washington,D.C. 20301-7100
8a. NAME OF FUNDING/SPONSORING ORGANIZATION — 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. — PROJECT NO. — TASK NO. — WORK UNIT ACCESSION NC

11. TITLE (Include Security Classification)

Eastport Study Group A Report to the Director, Strategic Defense Initiative Organizat

12. PERSONAL AUTHOR(S)

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM ______ TO ______	December 1985	72

16. SUPPLEMENTARY NOTATION

Technical Support by ANSER, Arlington, Virginia.

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	BM/C^3,Computing Resources,Software Technology,Architectur

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The Eastport Study Group is a panel from industry, Government, and Academia. The purpose of the study is to evaluate the Strategic Defense battle management problem, devise a response, and to recommend an appropriate research and development program to implement the response. The report stresses the approach of developing battle management and systems architecture together.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED [✓ SAME AS RPT.] ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Ensign James L. Coulter	22b. TELEPHONE (Include Area Code) (202) 653-0033	22c. OFFICE SYMBOL SY

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete. — SECURITY CLASSIFICATION OF THIS PAG

BLOCK 12.

Cohen, Danny
Information Sciences Institute
USC
Marina Del Rey, CA.

Chatterjee, Dr. Bijoy
Advanced Technology Directorate
ESL
Sunnyvale, CA.

Lau, Dr. Richard
Scientific Officer
Office of Naval Research
Arlington, VA.

Lipton, Dr. Richard J.
Professor, Dept. of Computer Science
Princeton University
Princeton, NJ.

McLaren, William G.
Maj. Gen. USAF (RET)
Vice President
V. Gerber Associates

Mizzel, Dr. David
Scientific Officer
Office of Naval Research
Pasadena, CA.

Seitz, Dr. Charles A.
Professor of Computer Science
California Institute of Technology
Pasadena , CA.

Smith, Dr. Alan J.
Associate Professor, Computer Science Division
University of California
Berkeley, CA.

Brothers, Walter L.
Senior Weapons System Analyst

Crossland, Anita L.
Editor

Stout, Dr. William L.
Senior Mathematician

Eastport Study Group

SUMMER STUDY 1985

A Report to the

Director

Strategic Defense Initiative Organization

December 1985

CONTENTS

ACKNOWLEDGMENTS

All the members of the panel contributed to this report by writing various chapters.

It was the dedicated work of Chuck Seitz that made this into a coherent report. Chuck's help ranged all the way from the top conceptual level to the editorial level.

Thanks to Major David R. Audley of SDIO-SY for helping us understand and address the problems.

Thanks to all the members of the Eastport group, and special thanks to Chuck for getting this report done.

Danny Cohen
Marina del Rey, 1985

EXECUTIVE SUMMARY

The SDIO Panel on Computing in Support of Battle Management was appointed "to devise an appropriate computational/communication response to the [SDI battle management computing] problem and make recommendations for a research and technology development program to implement the response."

The panel concludes that computing resources and battle management software for a strategic defense system are within the capabilities of the hardware and software technologies that could be developed within the next several years. However, the anticipated complexity of the battle management software and the necessity to test, simulate, modify, and evolve the system make battle management and command, control, and communication (BM/C^3) the paramount strategic defense problem.

Software technology is developing against inflexible limits in the complexity and reliability that can be achieved. The tradeoffs necessary to make the software task tractable are in the system architecture. As indicated in the Fletcher Report, the "applique approach" of designing the system first and then writing the software to control it is the wrong approach for SDI. System architecture and battle management must be developed together.

One promising class of system architectures for a strategic defense system are those that are less dependent on tight coordination than what is implied by the Fletcher Report. The advantages of this type of architecture include robustness, simplicity, and the ability to infer the performance of full-scale deployment by evaluating the performance of small parts of the system. The panel prefers an unconventional architecture that simplifies the software development and testing tasks over reliance on radical software development approaches and the risk that reliable software could not be developed by the "applique approach" at any cost.

(Blank)

I. OVERVIEW

The SDIO Panel on Computing in Support of Battle Management was appointed "to devise an appropriate computational/communication response to the [strategic defense battle management] problem and make recommendations for a research and technology development program to implement the response."

The panel met for 17 days during the summer of 1985. These meetings of the entire panel included extensive discussions among panel members and many useful interactions with SDIO management, SDI contractors, and outside software engineering experts†. Because the technical problems stimulated the interest and curiosity of the panel members, they have worked individually and in small teams between meetings, keeping in touch by mail, telephone, and computer networks. These additional activities included reading reports and articles relating to SDI, writing this report, consulting technical experts in a variety of fields, analyzing different ideas and approaches, performing small-scale computer simulations, and visiting contractors. All information that the panel as a whole discussed and the contents of this report are unclassified.

The panel members appreciate SDIO's effective way of instructing them to think broadly about technical issues and possibilities, and of providing a wealth of information ranging from broad concepts to architecture studies and current research details.

This panel's study follows almost two years after the report: "Battle Management, Communications, and Data Processing", Volume V of the "Report of the Study on Eliminating the Threat Posed by Nuclear Ballistic Missiles", James C. Fletcher, study chairman. We hereafter refer to this volume as the "Fletcher Report". This report was the panel's first reading assignment and starting point; it is a very insightful early study of many of the same problems that this panel has addressed.

In the interest of making this report more readable by persons who have not studied other reports on the Strategic Defense Initiative, this chapter starts with a general discussion of SDI and the problem of ballistic missile defense. The main body of this overview summarizes critical technical issues and the panel's recommendations to SDIO.

† The panel wishes to thank Bob Balzer, Carl Hewitt, Jim Horning, David Parnas, and Vic Vyssotsky. Particular thanks go to Vic Vyssotsky, who in his understated way provided many important insights to the software development task.

A. Introduction

From "The President's Strategic Defense Initiative", a White House pamphlet dated January 1985:

> "[SDI's] purpose is to identify ways to to exploit recent advances in ballistic missile defense technologies that have potential for strengthening deterrence – and therefore increasing our security and that of our Allies. The program is designed to answer a number of fundamental scientific and engineering questions that must be addressed before the promise of these new technologies can be fully assessed. The SDI research program will provide to a future President and a future Congress the technical knowledge necessary to support a decision in the early 1990s on whether to develop and deploy such advanced defensive systems."

SDIO planners consider the Strategic Defense Initiative a potentially very long-range effort. The current research phase is aimed at providing, during the next several years, the best information possible concerning the feasibility and cost of alternative new approaches to ballistic missile defense.

One purpose of this research phase is to provide the technical basis for an informed decision by a future administration and Congress whether to proceed to the development and deployment of a strategic defense system. A second purpose of this research is to provide a scientific base on which the engineering development can grow, should a future administration and Congress decide to proceed to a development. As is essential for any long-range effort, SDIO sees that the fundamental research must continue concurrently with all later phases of the initiative.

The panel does not expect small-scale and/or early technology deployments that might occur during the 1990s to provide a "near-perfect" defense. Rather, initial deployments influence our strategic position largely in their ability to intercept enough incoming warheads to enhance Soviet attack uncertainties. Possible Soviet countermeasures such as fastburn rockets or decoys also reduce the useful payload of their missiles.

At the rate at which the relevant technologies – sensors, weapons, computing, and communication – are developing, a strategic defense system of the 2000-2010 decade could provide a sufficiently effective defense that no Soviet planner could be reasonably assured of the "success" of a ballistic missile attack. The United States would then no longer need to depend solely on an offensive deterrence. Such a system, however, would still require us to upgrade our capabilities to reflect advances in the base technologies and changes in the threat situation.

The research is then directed toward a defensive system that is:

- Potentially long-lived.
- Based on exploiting advancing technologies.
- A step beyond existing military experience and doctrine.

SDIO accordingly should not and is not approaching the SDI in the same way that the DoD ordinarily "procures" research or initiates the development of a new tank, missile, or aircraft. Identifying promising technological opportunities is a part of the research problem; the research is expected to define the operational requirements for the components of a strategic defense system.

If developed and deployed, the longevity of a strategic defense system clearly affects the formulation and implementation of the computing system that controls it. The system must be capable of continuous evolution while in reliable service over a longer period of time than has been foreseen for other defense systems. This requirement, perhaps the most fundamental of all the system requirements, influences all of the technical issues subsequently described.

B. Physical Characteristics

The following review for the lay reader may be taken also as a summary of the panel's working assumptions about the physical characteristics of the strategic defense problem.

The physical dimensions of the battle management problem are well understood. A ballistic missile can first be intercepted during its *boost phase*, which lasts for only several minutes. During this phase the missile emits enormous amounts of energy with a distinctive spectral signature and is accordingly easy to detect and locate. The missile is also a relatively large and fragile target. When the boost phase is completed, the missile releases a *bus* containing on the order of 10 *reentry vehicles* (RVs). The bus then launches the RVs, each into a slightly different ballistic trajectory. Future buses developed in response to defensive systems can be expected to launch both RVs and decoys. Interception of the missile in the boost phase, or of the bus, offers the advantage not only of dealing with a larger target than an RV, but also of eliminating the many individual targets – RVs and decoys – that are launched from the bus of a single missile.

During the *midcourse phase*, the RVs and decoys follow a ballistic trajectory. This phase lasts for 20-30 minutes for intercontinental ballistic missiles (ICBMs). The ability to distinguish RVs from decoys is obviously advantageous, and SDIO is researching this problem. If the heavy RVs and light decoys are not correctly distinguished during the midcourse phase, whatever of them remain finally sort themselves out as soon as they start to reenter the Earth's atmosphere. The final opportunity for defense, the *terminal phase*, may last as little as 40 seconds.

The midcourse phase for submarine-launched ballistic missiles (SLBMs) may be extremely brief to nonexistent, so that the only opportunity to intercept them may be during the boost, postboost, or terminal phases.

The basic concept for the *system architecture* of a strategic defense system is to employ at least three *tiers* in the defense: at least one for each phase. Each tier must include sensors to locate the targets and weapons to destroy them.

The most likely technology for the weapons for the terminal phase defense are ground-based rockets that are capable of very high acceleration in order to intercept the incoming RVs at a high altitude. Weapons able to intercept missiles in the boost phase, however, must have global coverage and accordingly must be in orbit. The same weapon platforms used for the boost phase defense could also be used for midcourse defense, since the requirements on their geographical distribution and weapon characteristics are similar. The space-based weapon platforms might be augmented by additional ground-based weapons launched only after the beginning of an attack.

The orbiting weapon platforms, at an altitude of several hundred kilometers, might employ either *kinetic energy weapons* (KEWs) or *directed energy weapons* (DEWs). We will assume that the useful range of either of these types of weapons is about 2000km. This range is determined in part by weapon characteristics and in part by the line-of-sight from the altitude of the platform. The weapon platforms would be put into a set of orbits that are synchronized to each other (not geosynchronous), thus ensuring approximately uniform coverage of the entire Earth's surface at all times. Random orbits are not desirable because they produce predictable holes in the coverage.

The kinetic energy weapons are high-speed rockets or bullets that destroy targets by impact. These kinetic energy weapons appear to be the most feasible for early deployments. Their advantages include a lack of dwell time, which DEWs have, and their consequent rapid firing rate. There may, in other words, be many KEWs fired in a short interval from a single weapon platform. The disadvantages of KEWs relative to DEWs include their limited (expended) ammunition and the delay between firing and being able to assess whether the shot destroyed the target. The directed energy weapons do not appear to be candidates for early deployments but offer possible advantages when they are fully developed. None of these space-based weapons is capable of causing damage at the Earth's surface.

Some space-based sensor platforms would be located in high orbits; others in low orbits or combined with weapon platforms. The space-based sensors can be passive, detecting targets by their electromagnetic radiation, including their thermal radiation. Other sensors such as terminal defense radar could be located on the ground or in aircraft.

All the system resources would be tied together with a communication system and would operate under control of the battle management system. It is these battle management, command, control, and communication (BM/C^3) issues that we discuss in the later sections of this overview and report.

The fraction of our force that can be applied during the boost phase depends on the area distribution of missile launches. With a 2000km range, each platform covers about 2.5 percent of the Earth's area. The significance of this 2.5 percent figure is that the fraction of our force that could be used in the boost phase if all the launches were from one place is 2.5 percent. Curiously, the ideal deployment against a defensive system is concentrated, while the ideal deployment against an offensive system is dispersed. With a launch area as large as 1000 by 2000km, about 10 percent of our defensive force could be applied during boost phase, plus whatever could be used against SLBMs.

These numbers relating to the fraction of our forces that could be used are independent of the number of weapon platforms only if there are sufficient platforms for full coverage. Boost phase coverage in partial (early) deployments of weapon platforms would be nil. On the other hand, high altitude sensor platforms in early deployments have the potential of providing earlier warning of a ballistic missile attack than existing ground-based systems. Generally, even for a given number of weapons, it is better if they are distributed over more platforms, since the average range is then smaller, and the effect of one being destroyed or out of service is not as severe. Use of orbiting defensive forces during the midcourse phase is more favorable: 30-40 percent according to simulations.

Necessary accuracy and resolution in weapons and sensors is not phenomenal. A pointing accuracy of 1 arcsec is within a 5m error radius at 1000km. This same resolution is at the diffraction limit for green light with an 0.1m aperture. Objects – platforms and targets – moving at orbital velocities are moving fast enough that speed-of-light delays must be compensated. For example, when the electromagnetic signal from an object moving at orbital speed reaches a sensor, it is already 1m further along its path for each 40km distance to the sensor.

C. Battle Management Policy and Doctrine

Historically, the pattern used for the acquisition of weapons systems has been to acquire the weapons first and devise the command, control, and communication (C^3) as an accessory. The Fletcher Report made the point that this "applique approach" is the wrong approach for SDI. The panel strongly agrees; it regards the battle management and C^3 problem as the paramount technical issue that must be resolved in order to implement a strategic defense system.

Defining the military operational requirement (MOR) is one initial key to the architectural development. Clear definitions within the various descending levels of operational "needs" in the MOR are necessary at the outset. Simply stated:

- Either the end user, the military, states its needs and the architecture developer produces to those criteria, or

- Regardless of the operational requirement, the developer uses his assumptions to produce an architecture within which the end user is constrained.

If various teams of competing contractors were devising candidate architectures that satisfied a common MOR, the eventual selection process could start from a common basis of understanding.

Unfortunately, the task is not that simple. The military operational requirements for a strategic defense system – a system that has no clear precedent in military experience – have not been devised. The "problem" of strategic defense is imprecisely posed in the best sense: The objective is clear, and the exploration of means to achieve the objective is most appropriately conducted with many degrees of freedom. SDI research should gradually make the range of possibilities and solutions more precisely understood.

Although the panel has discussed possible strategies and doctrine, the answers to many questions involve considerations that are beyond the panel's expertise. For example, to what extent should one consider attacks against the defense system itself?

It appears that the SDI Phase I† architecture contractors used Volume V of the Fletcher Report as their "common baseline" by default. The panel observes that the technical problems of the system architecture and software development for a battle management system are interwoven with the entire problem of defining also – perhaps only after several phases or iterations – the MOR, policies, and strategies for its use.

This observation implies the need for flexibility in the architectural development. One can anticipate various transitional phases in any eventual deployment of the SDI system. One can visualize sensors of a defensive system that, at the time of their initial operational capability (IOC), would serve operational aspects of both offensive and defensive actions. Use of SDI sensors for warning of a ballistic missile attack does *not* imply an automatic response. In fact, it could provide additional time for the human decision-making process.

† The term "Phase I" refers to the first phase of the "System Architecture Key Tradeoffs Study" conducted by SDIO early in 1985.

As the defensive system expands in numbers, capabilities, and reliability, one can expect the offensive deployments to diminish. The evolving architecture and battle management system should be able to accommodate the changing nature of the force deployments.

While the system evolves, one should recognize that human intervention is necessary to control strategic operational forces. The same degree of control required for the strategic offense may not be required for a solely defensive force. Thus, the eventual process of political approval for SDI could lead to alterations in the military doctrine employed by the operational forces. The system will require continuous user-developer interactions to ensure that the architecture meets the evolving threat and the dictates of national policy.

D. Battle Management Principles

The most plausible organizations for a strategic defense battle management system are hierarchic. That is, their communication and information-processing structure can be portrayed graphically in "tree" diagrams such as an organization chart or depiction of a chain of command. This hierarchy or tree structure of a battle management system is rooted in the command authority and branches to the sensor and weapon subsystems. The properties of such hierarchic systems are very well understood both analytically and by analogy to this same organization having been adopted by living creatures and their social organizations.

Such systems have the property that information sensed at the "leaves" of the tree is processed (compressed) into more abstract representations as it is communicated toward the root, and is articulated from abstract representations to detail commands as it is communicated toward the leaves. This communication and computation structure preserves locality and allows for autonomous actions from local subunits, which have the same tree-like structure as the entire system. When the communications are augmented with lateral paths, the hierarchy can also be made fault-tolerant. The (superficial) variations on this basic architecture are defined by the number of levels and functional definition of the levels, particularly their capability for independent action.

This hierarchic organization determines the logical structure of the communications within the battle management system and accordingly strongly influences its physical organization. The hierarchic structure also influences the strategies for managing the complexity of the battle management system software.

One should regard the sensor and weapon parts of a strategic defense system as well-defined subsystems analogous to computer "peripherals". Although the sensors and weapons can be expected to have self-contained computational resources for tasks such as signal processing, aiming, message processing, and self maintenance, they themselves do not have any responsibility for coordinating their actions with those of other platforms or for allocating resources.

This model is consistent with the notion that the software for sensors and weapons could be largely conventional and tied closely to the technology and capabilities of the particular subsystem. For example, a sensor subsystem might have the ability to accept commands to search a given area for a particular signature. It could articulate this command into the suitable actions of pointing its sensors, performing the required image-processing operations, and reporting the results, not as an image, but as a set of measurements.

The battle management system and software are then defined as those parts of the system that deal with coordination. One can envision many different types of coordination, each with its own purpose, latency requirements, and performance benefit. Different types of coordination require more or less of the entire system to be involved, and therefore would be accomplished at different levels in the logical hierarchy. Examples:

- Lowest levels: It would be desirable for certain low-level coordination, such as stereo and other "sensor fusion" operations, to occur as close to the leaves of the hierarchy as possible.
- Middle levels: Target discrimination and attack coordination (assuring complete coverage, avoiding multiple shooting) might well occur at the middle levels of the hierarchy.
- High levels: The assignment of priorities of targets in midcourse in order to prevent particular areas from being overwhelmed in terminal defense, or to prevent any single area from accepting too high a concentration for terminal defense, requires global information that must be processed close to the root of the hierarchy.
- Highest levels: High-level command and control decisions defer to the root.

Although all of the panel's briefings have emphasized that the logical organization does not determine the physical organization, in practice there is every reason in such a distributed system to pattern the physical organization after the logical organization. In fact, with satellites in different orbits, one wishes generally to keep the system in a logical configuration that reflects the current geographical distribution of the satellites. In other words, the orbiting assets would not be regarded as a single system; they would be organized into "battle groups". Membership in the groups would change dynamically according to the patterns of the orbits and the serviceability of the platforms.

This distributed approach to battle management minimizes the communication latency and required bandwidth, as well as the consequences of interruptions of communication or loss of system resources.

E. Phase I System Architectures

Volume V of the Fletcher Report describes a battle management approach that should be interpreted as a "baseline" example that the computation and communication resources required for battle management are within reasonable technological expectations. This discussion of "Architecture for Battle Management" includes the caution that it "... is not intended as a design manual".

The Phase I contractors who made presentations to the panel† nevertheless followed the Fletcher Report approach very closely – certainly more closely than the authors intended.

Although much of the Fletcher Report discussion is centered around a global data base, including a track file of all objects, the report indicates that there are "technical issues needing attention in the design of systems that depart from the ideal of total information at every agent. It is important to know, for example, how well one can allocate weapons to targets using algorithms that depend only on local data. Not only would such algorithms be simpler and more easily mechanized, but their use would also directly affect the extent to which data must be distributed throughout the system". The panel has determined that those Phase I architectures presented were deficient because they fail to address these technical issues.

The proposed Phase I system architectures presented to the panel were developed around sensors and weapons. In spite of the sound advice in Volume V of the Fletcher report that "the battle management system and its software must be designed as an integral part of the ballistic missile defense (BMD) system as a whole, not as an applique", these contractors treated the battle management computing resources and software as a part of the system that could be easily and hastily added. The contractors treated battle management as something that is expected to represent less than five percent of the total cost of the system, and therefore could not significantly affect the system architecture. They have developed their proposed architectures around the sensors and weapons and have paid only "lip service" to the structure of the software that must control and coordinate the entire system.

† The panel received presentations from several contractors, who will remain unidentified here. We were told that SDIO regarded the studies produced by these contractors as among but not necessarily the best, and that they were representative of the diversity of approaches that the contractors had come up with.

As a result, none of the architectures presented to the panel adequately addresses the key issues of structuring the system architecture to (1) make the battle management software task tractable, and (2) accommodate testing and evolution of the system and software.

Similarly, SDIO should guard against the assumption that only the physical requirements implied by the sensor and weapon characteristics drive the system architecture. In fact, the battle management, system integration and evolution, and testing requirements have at least as much impact on the system architecture. The SDIO organizational separation of system architecture from battle management should not be allowed to keep these aspects of strategic defense from being developed together.†

F. System Architecture

The panel concluded that computing resources and battle management software for a strategic defense system are within the capabilities of the hardware and software technologies that could be developed within the next several years. The panel cautions, however, that the feasibility of the battle management software and the ability to test, simulate, and modify the system are very sensitive to the choice of system architecture. In particular, the feasibility of the battle management system software is much more sensitive to the system architecture than it is to the choice of software engineering technique.

Accordingly, the recommendations that the panel regards as most important concern the relationship between the system architecture and the feasibility of the battle management system software. Software technology is developing against what appears today to be relatively inflexible limits in the complexity and reliability that can be achieved. The tradeoffs necessary to make the software task tractable are in the system architecture.

As indicated in the Fletcher Report, the "applique approach" for the software is the wrong approach for SDI. The problem of reducing software complexity must be treated as the principal influence of the system architecture.

In short, the panel recommends a broader study of system architectures than is represented by the Phase I efforts, with the objectives stated previously. The

† In *The Mythical Man-Month: Essays on Software Engineering,* Addison-Wesley, 1974, Frederick P. Brooks, Jr. says: "Conway's Law predicts: 'Organizations which design systems are constrained to produce systems which are copies of the communication structures of these organizations.' Conway goes on to point out that the organization chart will initially reflect the first system design, which is almost surely not the right one. If the system design is free to change, the organization must be prepared to change."

panel much prefers an unconventional system architecture whose programming is within the anticipated limits of software engineering over reliance on radical software development approaches and the risk that reliable software could not be developed by the "applique approach" at any cost.

For example, a promising class of system architectures for a strategic defense system are those that are less dependent on tight coordination than what is implied by the Fletcher Report. There are several aspects that make this type of architecture promising for a strategic defense system. They are robustness, simplicity, and the *ability to infer* the performance of full-scale deployment by evaluating the performance of small parts of the system.

The Fletcher Report suggests as an "ideal" – a model not to be adhered to in an implementation – a very central architecture (*e.g.*, the track files). Such centralized systems tend to be monolithic and sensitive to software errors. The panel is convinced that the battle management system should be an open and distributed system that takes advantage of the strategic defense system's special characteristics, such as dynamics, size, and the lack of need for global consistency and synchronization.

In conventional large distributed systems there are important issues of global consistency and synchronization of all the data bases. For example, a typical banking system has a global *state* that reflects every transaction that occurs. Transactions are serialized in order to maintain a valid total state. A loss of even the smallest transaction can invalidate that "historic" state. A strategic defense system does not have to maintain either global consistency or global infinite response state. It should distribute its functions while maintaining only central command authority and global situation assessment.

The panel has discovered that tight coordination of the system assets in certain cases (*e.g.*, to achieve a complete coverage and to avoid "double shooting") provides relatively slight (≈20 percent) improvement in system performance and is a potential source of many technical difficulties in reliability and testability. Some forms of coordination are still essential, for example:

- Tight coordination is required for central command authority.
- Tight coordination is required, but only over localized zones, for processing multiple (stereo) sensor information.
- Loose (optional, or not time-critical) coordination is required for the selection of targets during the midcourse phase, based on discrimination information and avoiding concentrations in the terminal phase.

Future architectural studies should examine the consequences of reducing the assumed degree of coordination of system parts.

G. Testing

A deployed strategic defense system must evolve with technology and perceived threat. This requirement is also a necessary property of our strategic offense, the Triad, and of each of its components, as well as of high-technology defensive forces such as antisubmarine warfare. These systems are not only continuously upgraded, but are also subjected to frequent tests – whether technical tests or military exercises – that provide the assurance that they can be depended on if needed.

There is an obvious synergy between evolution and testing. One must realize, however, that short of a war, there can never be a full-scale test of a strategic defense any more than there can be a full-scale test of our strategic offense. There are two reasons why we can be confident of our strategic offense:

- *Independence:* The effects of the forces are believed to be additive. For example, if one particular missile has been tested 20 times and has been on target 16 times, we can assume that these numbers can be adjusted proportionately to reflect results of an actual attack. One missile's malfunctioning does not generally cause another's.
- *Diversity:* There is a useful diversity in the Triad and each of its components, so that even if the Soviets were to develop an effective countermeasure for one weapon or system, the others provide a credible deterrent.

A full-scale strategic defense system would not be expected to involve more military hardware than any one component of the Triad. What makes it different? A characteristic of a strategic defense system that could make it difficult to test to the point of full confidence is an excessive reliance on coordination between the parts of the system. Because coordination is a battle management system function, the requirement of testability falls most seriously on the formulation of the battle management system and system architecture.

Computing systems involving complex interactions, such as operating systems, may appear to work perfectly under light computing loads and ideal assumptions, but they may fail when heavier loads create unexpected combinations of delays and/or faults in the system or its peripheral parts. Simple oversights cause most of these failures, but some are due to the very large number of combinations of circumstances that must be anticipated in writing the program.

The ability to have confidence in a total strategic defense system is a critical technical point that must be addressed as a central objective of the battle management system and system architecture. There are several approaches to this testing problem that are promising:

- Develop system architectures in which there is relatively little dependence on coordination or in which the coordination information is used only if available. These architectures are relatively easier to test in parts and possess other advantages mentioned in previous sections.

- Use simulation extensively, using very high-speed and/or highly concurrent programmable computers that would allow the operational code and algorithms to be tested under very large numbers of battle variations. It should be possible to employ such simulations to aid in debugging and in performance evaluation under varying tactics.

- Develop in-line† testing of a deployed system, with simulated data going to and from the sensor and weapon platforms. Hierarchically organized systems lend themselves readily to this type of testing. Such testing should be employed both on the ground and as an ongoing process in a deployed system.

Again, one must consider that the tradeoffs required to make a strategic defense system testable are found in part in the formulation of the system architecture. In this case the ability to infer by testing parts of the system the performance of the entire system is analogous to the case of the strategic offense.

H. Simulation

Simulation is vitally important to two aspects of SDI research and development. First, simulation is necessary to explore the way in which a variety of system architectures perform against many possible attacks and under different circumstances involving system flaws and component failures. Simulation is an excellent vehicle to support such design studies. Second, simulation allows subsystems to be tested while they are under development, and many of the same techniques can be extended to in-line testing of a deployed system.

A simulation system itself consists of computers, simulation language software, and simulations of weapons, sensors, and communication channels. In a typical simulation, the simulator monitors and computes the physical evolution of the "battlefield map" while interfaced to prototype or actual BM/C^3 software and attack scenarios. An essential and often overlooked component of simulation is validation – the effort required to ensure that the simulation is realistic.

† "In-line" refers to the technique of testing components and system in their actual operating environment rather than on a "test bench".

The panel recommends the implementation of several simulation systems for evaluating battle management architectures and strategies. Multiple simulations provide cross-validation, verification, and a higher level of confidence in the results. Further, the panel recommends the use of independent validators whenever possible, including efforts by research and/or academic groups.

The panel recommends also the implementation of several detailed simulators capable of supporting BM/C^3 software in three forms: simulated, prototype, and actual. These simulators would serve initially for debugging and simulated testing; later they would provide in-line tests of the system through its ability to interface with real sensors, communication channels, and weapons. The simulators should be capable of multilevel simulation in which different aspects of the system are represented at different levels of detail.

These simulators should be able to cooperate in large, distributed system simulations. The simulation facility should not be centralized at and restricted to the proposed National Testbed (NTB). The simulation effort will benefit significantly in quality and credibility if it is not "kept behind walls". The simulators should have extensive facilities for measurement, monitoring, and debugging. Again, one of the reasons for having a diversity of simulators is for establishing verification and cross-validation.

Battle management simulators should be developed early in order to evaluate BM/C^3 system designs, and the results of those studies should be incorporated into ongoing architectural efforts.

I. Software

The panel has discussed software technology issues extensively. The panel has also discussed this subject with experts in the software field.

Simply because of its inevitable large size, the software capable of performing the battle management task for strategic defense will contain errors. All systems of useful complexity contain software errors. Therefore, for the very high level of reliability required for a strategic defense system, the central question is how to design this system such that errors are first minimized and then tolerated.

In evaluating the impact of software errors, one should distinguish between errors that are system-critical and those that are not. For example, the Shuttle software has errors that become famous whenever a launch is delayed, but the Shuttle has never encountered *any* error in the critical ascent/descent code.

Techniques for constructing systems that are usefully reliable in spite of imperfection is not in the current mainstream of software engineering research. Some practitioners of software engineering are more concerned today with making their discipline resemble mathematics. There are, however, many techniques that permit errors to be minimized and their effects tolerated. Many of these approaches have been used before; some have only been discussed but not seriously applied to projects due to time and cost limitations; some are research topics.

The panel advises SDIO not to expect any single software technology to be *the* solution – the "magic bullet" – for the SDI battle management system. Instead SDIO should use proven software technologies as they fit each part of the problem. For example, the panel does not expect automatic programming, program verification, artificial intelligence, or knowledge-based expert systems to be sufficient in themselves.

The panel expects the realistic approach to the battle management software to be an open, distributed, adaptive self- and cross-checking system with extensive redundancy.

J. Hardware

Unlike the software technology, where progress has been and will likely continue to be relatively slow, the panel expects that hardware technology will continue to progress rapidly. Advances in both devices and architectures are leading to systems in which increased computing speed is combined with reductions in size, weight, and power. This progress is a result of ongoing research in the commercial sector and government-sponsored efforts such as the VHSIC program of DoD and the VLSI and Strategic Computing programs of DARPA.

The panel therefore recommends that the SDI exploit the advanced computer technologies to simplify the more difficult tasks such as software development and testing. The objective when acquiring hardware should be to determine how faster computation can simplify a task, rather than how a task can be performed with minimal computing capabilities. Initial review of the Phase I architectures revealed conservative use of computers in all areas.

A very important area for immediate attention is the SDI space qualification process. Technology insertion delays are significant in the space applications. Better understanding of the space environment and of behavior of hardware in space is now available. This knowledge together with progress in the design of VLSI circuits using innovative design tools provide an excellent opportunity to deliver the current technology in space.

It will be necessary to propagate a different culture of system development to exploit the emerging technologies. Insertion of technology requires a continuous upgrading of methods that are consistent with new techniques. The demands of project schedules, lack of capable staff, procurement decisions, and bureaucracy have created an industrial culture that resists change. We must create a culture that is able to adapt to changes more effectively than we currently do. This objective requires that we study causes and develop practices that encourage and reward responsible effort toward technology insertion.

K. Communication

Another unnecessary architectural assumption derived from taking the Fletcher Report beyond its intention is the form of networking of the system assets. Using the sensor and weapon platforms also as the communication backbone rather than considering a network of satellites dedicated to communication could negatively affect reliability, flexibility, and cost of deployment.

The communication approach proposed in the Phase I studies reflects a misperception of the issues that drive the solution. If the communication approach is driven by the hardware rather than by the problem, the solution focuses on data transmission rates rather than on information exchange. This approach has already focused attention on the transmission capability and ignores packet processing, encryption/decryption, and error correction, in spite of the existing disparity between feasible rates for raw transmission and packet processing.

SDIO should study and develop communication networks that address the specific and unique requirements of the strategic defense system.

The panel recommends also that SDIO build an "SDInet" to serve as a proving grounds for the distributed system architecture of the strategic defense system and for its communication protocols. The SDInet will support interoperability of architecture work, battle management development, and simulation facilities. The SDInet will also support resource sharing among the SDI contractors.

L. Program Management

The scope and rate of advance of the technology areas associated with the SDI BM/C^3 investigation presents SDIO with a large and complex management problem. Such a program clearly requires technical direction, progress review and evaluation, and effective program management and contracting practices. The special characteristics of the program, particularly its dependence on advancing technologies, justify innovative approaches to program management.

The panel recommends that SDIO use independent contractors to perform the technical functions of program managers in those areas in which SDIO can not enlist personnel of the appropriate technical caliber. This use of contractors would allow SDIO to tap the talent of leading researchers in the scientific community.

An SDI technical infrastructure should be developed to provide a resource-sharing mechanism and to promote the creation of an SDI technical community. The SDI contractors should be linked together via SDInet, a computer communication network. This network would allow contractors to exchange technical information and software; it would promote compatibility and cooperation.

SDIO should sponsor research in the areas of program management and contracting practices. The program directed by SDIO must adapt to a rapidly evolving and changing environment. As a result, SDIO needs a program management structure and contracting method that allow it to alter and adapt its program as rapidly as technology issues are resolved. Automated means of tracking program interdependencies are needed, as are special contracting methods, or changes in specific DoD program management guidelines.

II. DOCTRINE AND POLICY

Generally, developments in weapons technology have led to operational concepts and a body of military doctrine that governs the use of those developments. In similar vein, the pattern used in the United States for the acquisition of weapons systems has been to acquire the weapons first and devise the supporting command, control, and communications (C^3) structure as an add-on. For the SDI, the panel has determined that the opposite is necessary – that the battle management system and its supporting C^3 must become the primary developmental factors. Thus, SDIO should develop the strategic defense C^3 first, then add the weapons and non-C^3 sensors, which one could call "peripherals" at this stage of the research effort. The Fletcher Report stressed that a C^3 "add-on" or "applique" approach is the wrong approach for SDI.

In addition, the need for a definite statement regarding the military operational requirement (MOR) is the initial key to the system's architectural development. Clear and definitive statements within the various descending levels of operational "needs" in the MOR are necessary from the outset. Reduced to basics:

- The user states his needs and the architectural developer produces to those criteria, or
- The developer uses his own assumptions to prepare an architecture within which the user is constrained, regardless of the operational requirement.

If the competing contractor teams were devising candidate architectures that could satisfy a common MOR, the eventual selection process could start from a common basis of understanding. The panel can not determine whether this specific process of definition took place in the Phase I and the Phase II architecture development projects. It appears, however, that selected contractor teams used Volume V of the Fletcher Report as their "common baseline" in the absence of a specific MOR.

A. Flexibility in Architectural Development

Weapons technologies evolve in response to an adversary's potential threats. So, too, does the architecture for the SDI BM/C^3 system depend on the forecast threat and how that threat evolves many years ahead. It is difficult for national intelligence-gathering organizations to identify a range of future threats with any detailed degree of accuracy. Thus, for both weaponry and the supporting BM/C^3, understanding the threat becomes an evolutionary process that is subject to changes and refinements on a continuing basis.

In a process where the threat and the ability of future technologies to cope with the threat are both unclear, the development of weapons and C^3 takes on a greater aura of risk. The development process requires flexibility to prevent stifling advanced weaponry research. Also, it is difficult to "freeze" an architecture early in a research program without potentially hindering the program's results. Specifying an architectural solution reduces developmental costs and risks as it structures the approach. The specificity, or "freezing", process, however, can not accommodate radical changes in the military posture of the defensive forces or the adversary. The combination of these factors leads to the conclusion that there must be an inherent flexibility in the architectural development. This flexibility will require a close, continuing relationship between the eventual user and the architectural developer.

The problems that a BM/C^3 system faces are not only those of accommodating the research phase and forecasting a fully deployed system. The BM/C^3 architectural developer must also consider a wide range of options for deployment. With any weapons system there is a "phase-in/phase-out" cycle of new weaponry for old. Establishing the spaceborne segments alone will probably encompass several years. Therefore, one can expect to have various transitional phases in any eventual deployment of the SDI system. One can also envision an evolutionary change in the strategic force mix as the SDI system begins its initial deployments and obsolete weapons are withdrawn from the active system's inventory.

In the time between initial operational capability and full deployment, a variety of procedural changes would occur in the nation's strategic command and control system. The BM/C^3 system for SDI must necessarily be interoperable with other C^3 systems that the strategic forces use. Throughout the deployment phase, the C^3 systems and their supporting architectures must be sensitive to unforeseen technological factors and political realities at international levels. Thus, the evolving architecture and the BM/C^3 system to support an operational SDI deployment must be designed to accommodate the changing nature of force deployments.

One should recognize from the start that this evolution requires a greater degree of human intervention to control residual strategic operational forces – a degree that might not be required for controlling an SDI force solely. These differences in the required degree of man-machine interface carry great significance due to the imperative to control these tremendously destructive forces.

Because SDI shifts strategic weapons from an offensive to a defensive approach, its critics and proponents have commented that various aspects of a future strategic defense system will affect national policy, military doctrine, and operational/procedural mechanisms that form the basis for the battle management.

The term "national policy" implies a consensus that represents the views of broad segments of the American public. A policy of this kind is necessarily more broadly based politically than are the goals of a particular political administration. Further, national policy is an evolutionary process. That is, national support changes with the goals of a revised strategic posture for the United States.

Military doctrine encompasses the codified body of knowledge that establishes, in various levels of specificity, the military means to carry out the national policy. The command and control mechanisms - or battle management - and the operational military process must be responsive to the changes in policy and doctrine.

Thus, the eventual political approval process for SDI could lead to alterations in the military doctrine employed by the operational forces. There could be a series of user-developer interactions to ensure that the evolving architecture meets the evolving threat and the dictates of national policy. Such an iterative process would have a "trickle-down" effect on the operational forces and the BM/C^3 system that those forces use at the lowest echelons.

The considerations outlined previously will also affect the design criteria, performance, and degree of simulation/testing necessary to ensure "acceptable" levels of compliance with the program's operational goals. To accomplish this broader task, the great number of technical factors woven throughout the program would require a total systems concept from the start.

B. Program Consistency and Budgetary Strategy

Several factors create the inconsistency of funding for C^3 programs that has long been a problem within DoD. Support for specific programs by an individual military service often has varied yearly and, on occasion, has borne little relationship to a previously agreed-upon joint position. Another factor involves the inherent instability of assignments of individual program managers and their staff superiors. Also, the vagaries of the annual defense and federal budget processes create budgetary fluctuations. Historically, the government and contractors have underestimated program costs for both hardware and software.

C^3 – as a generic item in DoD – has enjoyed relatively large increases in the levels of appropriated funding in the past four years. Budgeting is now at a level where any additional increases, even for SDI, become problematic. The SDI apportionment for BM/C^3, important as it is, shares the vulnerability of all other Defense C^3 programs to congressional fiscal reductions and the need to balance major budget programs.

Operating within the DoD Planning, Programming, and Budgeting System (PPBS), the creation of the young SDIO caused an irregularity among more established and "regularized" means of funding. The highly structured budget system of DoD requires almost full-time work of some members of the military services and defense agencies. The SDIO staff is too small to continually devote specialized attention to both details of the technical research programs and routine budgetary processes that permeate Pentagon life. In short, SDIO simply does not have the personnel to perform requisite tasks.

If SDIO is to promptly fund and award research contracts on aspects of BM/C^3 and accomplish research in a timely manner, the budgeting process – with all of its PPBS aspects – must become more "normalized" for long-term funding. That is, DoD must formally recognize the unique functions of SDI within the PPBS. This would establish and enhance relationships among the individual military services, the Joint Staff, and other DoD functions.

Developing such a normalization process will require innovative, unique, and specialized changes regarding the relationship of SDIO, its funding authorizations, and the military services – the product's ultimate users.

Recommendation: Establish an early dialog among the developers of the strategic defense system architecture, the developers of the battle management system, and the ultimate users of the system. It is the ultimate users of the system who should develop the military doctrine.

III. ARCHITECTURE

The Fletcher Report presents a *model architecture*† based on an assumption of each sensor and weapon having all situation information that may be relevant to its function. This model served as a baseline for examining the computing and communication demands of a strategic defense. The Fletcher Report clearly distinguishes between the role of such idealized architectural models and practical implementations.

The development and characterization of practical architectures for a strategic defense system is a complex endeavor. The "design space" of possibilities is very large, and there are many criteria by which to evaluate a given architecture. The following criteria are fundamental:

- *Performance:* There is no simple measure of the performance of a strategic defense architecture, in that it must be designed to respond to many different attack scenarios. Generally, however, the number and fraction of warheads intercepted in an all-out attack is a useful rough measure of performance. Other properties of an architecture, such as *reliability, durability, robustness,* and *diversity* contribute to its performance.

- *Testability:* The testability of an architecture can be measured by the *confidence* with which one can determine the performance of a deployed system. Because full-scale tests are impossible, just as they would be for the strategic offense, the system must be structured in a way that allows its performance to be inferred accurately from small-scale tests.

- *Cost:* The cost of a strategic defense system is of obvious relevance, and includes components of the cost of developing, building, deploying, and maintaining the system. Although the cost of building and deploying sensors and weapons is expected to dominate the cost of a deployed system, the *complexity* of the battle management software to control a given architecture transcends simpler cost measures. Excessively complex software can not be produced at any cost.

Earlier this year there were 10 contractor studies performed in "Phase I" of the "System Architecture Key Tradeoffs Study". Those of these studies that the panel has reviewed have adhered closely to the Fletcher Report. In particular, the

† See Volume V of the Fletcher Report, section 4.2, for a discussion of the utility and limitations of this model.

Phase I studies have focused on sensor and weapon requirements. The detailed analyses and simulations in the Phase I architecture studies provide useful insights into alternatives in weapon characteristics, numbers, and distribution. However, it is the panel's judgment that the Phase I contractors did not pay enough attention to the issues of software complexity and testability.

This deficiency in the Phase I studies is understandable, but must be corrected. On one hand, if the United States can not build sensors that can detect and locate the targets, or if it can not build weapons that can destroy them, then the software and testing requirements are purely academic. On the other hand, if the United States can not build computing systems and battle management software to control the sensors and weapons, then the sensor and weapon characteristics and placement are purely academic.

SDIO must not assume that any architecture with sufficient sensors and weapons in the right places is also a *feasible* architecture, *i.e.*, one that can be implemented successfully. While most of the architectures proposed – including the model architecture presented in the Fletcher Report – are sufficient in terms of sensors and weapons, the panel does not regard them as feasible. These architectures are not likely to be implemented successfully because they demand excessively sophisticated software and can not be adequately tested. However, the design space of possible strategic defense architectures is vast. It is the panel's firm conviction that there are architectures that have excellent performance and cost characteristics, and are testable.

A. Discussion

1. Coordination

The September 1985 issue of the IEEE *Spectrum* is largely devoted to articles about the Strategic Defense Initiative. The discussion of battle management begins with the sentence:

> "A Star Wars defense system would need to respond to an offensive strike as a single organism, coordinating perhaps millions of separate actions in a schedule timed in milliseconds."

The panel does not agree with this statement, particularly with the concept of the "single organism" coordinating every aspect of the system operation. For example, an army can not and need not coordinate each action of every soldier through the commander-in-chief. Instead, responsibility and authority is delegated in the chain of command.

Similarly, a strategic defense system need not and should not be tightly coordinated. The system architectures that the panel regards as the most promising are those that are organized as a *hierarchy* that is analogous to the military chain of command. It is important to understand that coordination is not *absent* in such a system. Rather, lower-level tasks are *delegated* to and *localized* within parts of a system. The parts can in turn delegate subtasks to their parts. The battle management decisions can then take place in a highly decentralized framework. No system part within such a hierarchy (see section I.D) needs to depend on millisecond-by-millisecond detail instructions from a higher authority.

The principle that can be applied to all forms of coordination within the spatially distributed strategic defense system is this:

Coordination exhibits the property that the time that can be allowed to accomplish the coordination increases as the distances involved increase.

The fundamental reason for this property is that the targets move very slowly compared with the communication signals. For example, the details of what may be going on outside of the "action radius" of a weapon platform is information that is of no immediate use to this platform.

Consider, for example, a small *battle group* formed from several sensor and weapon platforms that are within a few hundred kilometers of each other and a few hundred kilometers above a missile launching area at the beginning of a full-scale attack. The battle management local to that group may involve the exchange of a large volume of information about the location of the missiles as viewed from each sensor. The measurements made by the sensors are usefully combined or "fused" in the group's battle management system in order to provide more accurate tracking of the missiles than could be accomplished from a single sensor. This information pertains to a relatively small area, and might be updated every 10ms or so.

A condensed picture of the local situation would be reported for purposes of threat assessment to higher authorities in the hierarchy. The higher-level battle management combines the threat assessment information from many such battle groups and from high altitude sensors to present a condensed threat assessment to the command authority. Since this coordination involves the collection and processing of information – fortunately already condensed – from a large area, one cannot expect it to be updated more than every second or so.

Another type of coordination that could occur within a battle group is the assignment of weapons to individual missiles during the boost phase. The basic idea is to optimize the assignment based on weapon and missile positions, and also to avoid expending multiple shots at one missile while others are ignored. How important is this particular form of coordination? The panel has studied the possibility that weapon platforms attacking missiles during the boost phase choose their targets independently based only on local sensor data. Results of a preliminary analysis and simulations indicate that this decentralized approach would require only about 20 percent more "shots" to destroy the same number of targets than would a *perfectly* coordinated system. The tradeoff is that perfect coordination saves a certain amount of hardware at the cost of increased software complexity and decreased testability.

Coordination in the midcourse phase involves larger areas, more platforms, and more targets. The optimizations that could be applied are more complex, but there is more time available to perform these computations. The terminal phase of the defense again involves a multitude of smaller areas and the need for faster response.

The panel has concluded that the coordination required for each phase of the defense could use a decentralized and loosely coordinated approach to battle management. This approach does not incur a significant loss in performance compared with a hypothetical and infeasible model of a perfectly coordinated defense.

2. Decentralized Architectures

There are many critical advantages to an architecture that is decentralized to elements that are capable of independent action:

- *Simplicity:* This type of architecture can reduce the complexity of the battle management software by eliminating needless coordination. Since this approach does not require a globally consistent data base of all targets, it reduces the communication bandwidth and latency requirements, and relaxes scheduling demands.

- *Testability:* An architecture in which the elements – whether platforms or battle groups – are capable of independent action is testable. The idea is simple: Test each independent platform or group separately. If each works, then its independence allows one to infer that the whole will work also. This is exactly the same type of reasoning that allows one to believe that our highly complex strategic offensive force would work if called upon: Each missile, once launched, is an independent entity.

- *Evolvability:* System architectures that are not highly dependent on close coordination are more easily evolved to incorporate new technologies or to respond to changes in the threat. Such architectures also "scale" well from small- to large-scale deployments without requiring increased computing capability in each element.

- *Robustness:* Errors in one platform or battle group would have only a "local" effect: If one system part experiences an error, then only that part is affected. With a highly centralized architecture, the entire system could be adversely affected.

- *Diversity:* Decentralized systems also gain robustness through diversity. Any realistic plan for an evolutionary strategic defense must allow for a diversity of vendors and technologies and must accommodate continuous changes and updates to the system. If the architecture is composed of elements that were built and maintained by different vendors, each could employ different hardware and software within the same system protocols. Errors or vulnerabilities in one system are not likely to be duplicated in other systems.

- *Durability/Survivability:* The loosely coordinated system is more durable in the presence of hardware upsets and more survivable in the event of active countermeasures against the defense system itself. For example, if a platform loses its data base due to a flux of high-energy neutrons from a nearby nuclear explosion, unaffected platforms continue to function while the affected platform reconstructs its data base from freshly arriving sensor data.

3. Implementation Issues

Decentralized system architectures raise several interesting issues that the panel has considered and commends to the attention of system implementors.

First, there must not be a loss of human control over the independent groups. It might seem that the decentralized system implies a loss of human control over an "autonomous" entity. However, it can be made as difficult for a single battle group within a hierarchy to obtain weapon release authorization as it is for the entire system. The independence of the battle groups might even provide greater flexibility. For example, the command authority could authorize only some of the independent groups be armed in order to match the defensive level to the type of attack.

Second, coordination and communication are *reduced* in decentralized architectures; they are not eliminated. Some forms of coordination are sufficiently important to system performance to justify their inclusion, even if they are quite expensive in computation and communication. Consider, for example, the projection of ballistic trajectories in midcourse and assignment of priorities to targets in order to prevent any single area from receiving a high concentration of RVs. This form of coordination appears to be a valuable strategy for the defense. It can provide useful tracking information for the terminal defense, and substantially reduce the total number of warheads that "leak" through the defense. This form of coordination is difficult to test directly; one would have to rely on simulations to test this software.

Third, the decentralized approach might *appear* to be costly. Since the costs of the sensors and weapons so thoroughly dominate the cost of a strategic defense system, it could be argued that one must not "waste" shots; hence, the need for tight coordination. In every estimate of the total system cost, the most grandiose estimate of the software is less than a few percent of the total cost. This estimate could tempt designers to save hardware costs by writing more clever and complex software. The panel cautions that SDIO must consistently resist such temptations. The priority given to software development must be based on minimizing the software's difficulty. If this is not done, the United States could end up with a "cheaper" system that simply does not work.

For example, it may be possible to design architectures that use abundant computing cycles to reduce the software's complexity. In general, the most error-prone part of software stems from trying to optimize its performance. If simple, "brute-force" algorithms rather than complex, special case-laden methods are used, then the software is likely to be more reliable. It may also be possible to use additional computing cycles to avoid other complexities. For example, faster signal processing may be able to simplify the amount of information that must be passed from one sensor to other in order to achieve sensor fusion.

Finally, the panel has not been able to form a good quantitative argument on the tradeoffs between the number of platforms and their size and cost. Should there be 40,000 platforms each weighing 100kg, or 400 platforms each weighing 10,000kg, or something in between? It seems intuitively clear that survivability is enhanced by employing a larger number of smaller platforms. One could make the radar cross-section of the small platforms very small, but one can not effectively hide objects in Earth orbit due to their emission of $\approx 300^\circ$K thermal radiation against the $\approx 3^\circ$K sky background. The presence of more platforms may also favorably affect performance by allowing the platforms to be closer to the targets. Large numbers of small platforms may be cheaper to manufacture. SDIO should study this question.

4. Open Systems

Another important issue to which any architecture must attend is the ability to adapt to changes. These changes may be driven by corrections or by responses to possible countermeasures. Clearly, future enemy countermeasures are difficult to predict with great confidence. Also, first-time sensor surprise phenomena are commonplace: New sensors do not always behave initially as expected.

The panel recommends that the architecture for the strategic defense system be formulated as an "open system" that can allow the rapid insert of unanticipated new components and modification of existing pieces.

SDIO could begin with the design of a core of communication and information exchange protocols and build the rest of the strategic defense system around this core. In this way adding a new sensor or weapon to a group would be a relatively straightforward change to part of the system.

Designing open systems is still very much a research topic. One can, however, find the more rudimentary concept of providing a framework of standardized hardware and software interfaces in systems ranging from personal computers to advanced operating systems. This approach has, with remarkable consistency, been very successful when applied to such commercial systems. The panel expects that similar advantages are possible for an open strategic defense architecture, and therefore the panel recommends that SDIO should sponsor research into the design of open systems.

B. Recommendations

The panel recommends that SDIO:

- Develop a decentralized strategic defense architecture.
- Continue Phase I as an ongoing activity.
- Develop the battle management system as an open system that allows rapid insertion of new assets and modification of existing pieces. SDIO should also sponsor research into the design of open systems.
- Study architecture tradeoffs by early coupling of the architecture work with the development of battle management and simulation.

C. Summary

The most important point that the panel makes is that in evaluating any architecture for SDI, one must consider both the ability to make the required software work and the ability to test it. An architecture that can not be tested or that relies on software that does not work is of no value. This issue is so important that SDIO must be prepared to make a variety of tradeoffs among all elements of a strategic defense system to make the software and the testing tractable.

Finally, the panel stresses that a time-driven choice for a specific strategic defense architecture does not exist. The design space is vast, and a great deal of work remains before a clear choice can be made. Clearly, SDIO must evaluate architectures on their performance, cost, and testability before selecting the preferred strategic defense architecture.

IV. SIMULATION AND TESTING

Simulation has in recent years assumed a critically important role in research and engineering projects ranging from new computer designs to spacecraft. Simulation has an even more central role for SDI. Although it is possible for SDI to test weapons and sensors individually, the possibilities for testing a complete strategic defense system – therefore its battle management system – are limited. If carried out with unprecedented thoroughness and finesse, simulation can largely (but not completely) substitute for traditional testing.

In addition to the long-range importance of simulation in the engineering development of a strategic defense system, the panel expects simulation to provide the answers to important technical questions on which a development and deployment decision will be made. For example:

- Is it possible to design, implement, and test the BM/C^3 software for the strategic defense system so that it will achieve the necessary level of performance and reliability?

- Are the battle management strategies embedded in the BM/C^3 software adequate to cope with possible/potential attacks?

Conclusive answers to these questions require that the simulation facilities have the capability of modeling the components of the strategic defense system, and its potential threats, in great detail. These components include the sensors, weapons, communication, and battle management of the defense system, as well as the elements of a wide variety of possible attacks. The models must include the kinematics of the objects, the generation and propagation of the signals and communications that pass between them, and the component's internal computations and resulting actions.

The simulation facilities must also have the ability to replace simulated functions, whether hardware or software, with real components, for simulation validation and for testing purposes. This technique is referred to as *in-line* testing. Properly conducted, in-line testing can accelerate engineering development by allowing the insertion of system components and prototypes into a realistic system environment, thus detecting in early tests problems that would by traditional methods appear only in the system integration phase. In-line testing should continue to be used in deployed assets of a strategic defense system. For example, one could test the battle management functions of a "battle group" by communicating simulated data to the sensors and simulating the effects of weapons.

The advantage of simulation over testing is its ability to consider many cases, including situations that would be difficult, provocative, expensive, or impossible to duplicate in a test. The advantage of testing over simulation is its realism. The panel has emphasized elsewhere in this report the necessity of structuring the system architecture so that one can infer the performance of a full-scale deployment from a much smaller-scale test. Simulation in conjunction with testing has a similar purpose but is also capable of modeling the system operation at variable levels of detail.

The proposed simulation and test facilities will serve to:

- Simulate BM/C^3 architectures and battle management strategies to determine their performance when faced with various attack scenarios, and their continued performance (robustness) in the face of failures due to system flaws (bugs) or hostile action. Such simulations are the ideal vehicle with which to study system tradeoffs.
- Support prototype and actual BM/C^3 software to provide a base for the testing and debugging of the software as it is developed.
- Interface with real weapons, sensors, and communications channels to support in-line testing of the strategic defense system prior to and after deployment.
- Provide a facility for testing and evaluating ongoing improvements and updates to the BM/C^3 software.

The simulation facilities are essential for the designing, coding, and debugging of the strategic defense battle management system and will provide an important part of the basis for the system's credibility.

This chapter describes the functional requirements of simulation and testing facilities. It also outlines development projects needed to determine and meet system requirements.

A. Discussion

1. Simulation Facilities

The simulation facilities have functions that require operation at several levels. Testing and debugging the strategic defense system requires a simulation that runs with great detail and can support actual battle management software and interfaces to real weapons, sensors, and communications platforms. Evaluating BM/C^3 strategies requires rapid but less detailed simulations of large numbers of battles.

We expect that low-level simulators used for sensors and weapons development might run orders of magnitude slower than real time.

The mid-level simulators must be able to interface with other simulation systems or with components of a real strategic defense system to permit in-line testing. The panel envisions one or more of the mid-level simulation systems as distributed, with parts of a simulation running at various ground sites and parts running on platforms in space. Such simulators must necessarily run in real time.

The high-level simulators for studying BM/C^3 strategies could reasonably run very much faster than real time.

It is attractive to consider multi-level simulations in which different parts of a simulation are at different levels of detail. However, this idea is not very practical because of the orders of magnitude differences in speed. Thus, there is a need for careful abstraction at the different levels to produce and validate models of the system components.

Simulation efforts should cooperate closely with activities at the National Test-bed (NTB). But for several reasons, NTB should not be the only simulation facility. First, simulation is too vital to the strategic defense effort to permit a "monopoly" that would be implied by such a single implementation. Second, comparison of results from different simulation facilities, for purposes of cross-verification and validation, is essential. Third, a centralized implementation at NTB would likely lead to classification, very limited access, and narrow focus, which would curtail the simulator's effectiveness and prevent it from producing confidence in the system's functionality and reliability.

Therefore, the panel recommends that multiple simulation facilities be constructed for testing and debugging the strategic defense system and for evaluating BM/C^3 strategies.

Common interfaces should be used so that a simulator of one type can interact with one of another type and, if needed, could be interchangeable. Each could also verify and update the other.

Sensor, weapon, and communication channel interfaces must be defined. Simulation systems interfaces also must be defined so that parts of the strategic defense system can be simulated independently and on different computer systems that can be developed by different contractors.

In addition to the common interfaces mentioned previously, the various simulators should use the same abstractions and models such that similar results could be expected when similar input is given to different simulators.

The coordination among the various simulation activities to assure the commonality of the interfaces, protocols, abstractions, and models is an activity that must be performed explicitly. SDIO should recognize the coordination task of the various simulation activities as an explicit, independent activity.

The various simulators should be available for operation over a computer communication network, such as the SDInet discussed in Chapter VII (Communication), in order to provide easy and frequent access to their users.

2. Simulation Verification and Validation

Simulation *verification* ensures that the simulation system meets its specification and that its code is correct. Simulation *validation* ensures that the simulation meets the user requirements and corresponds with the real system it is intended to model.

SDIO should support research investigating the general problem of simulation verification, and in particular the actual verification and validation of the simulation development for SDI.

Each simulation component should be verified and validated by at least one *independent* party. The panel expects that the construction of multiple simulation and test systems will greatly aid the verification and validation process. Cross-comparisons among the facilities should reveal errors in the systems' specification, design, and implementation.

3. Weapon, Sensor, and Communication Simulations

The simulation of each weapon, sensor, and communication component should start out as a high-level representation and become refined in the level of detail as the actual component is defined and developed. Simulations should be developed jointly by simulation specialists and the contractors developing the actual systems.

The sensor simulators should access the battlefield map and produce a partial and distorted map that depends on the modeled capabilities of the simulated sensor, such as the detection probability as a function of range, angle, and signature.

The weapon simulators should access the battlefield map and produce hits that depend on the modeled capabilities of the simulated weapons, such as the hit probability as a function of range, angle, signature, and size of the target.

The communication simulators should deliver messages between the simulated sensors, battle managers, and weapons according to the modeled capabilities of the communication systems, such as error probability as a function of range, message length, and environmental factors.

4. Battle Management Strategy Evaluation and Testing

The primary function of the BM/C^3 architecture simulator is to evaluate different strategic defense system architectures and various battle management strategies against a variety of attack scenarios. SDIO should implement early the high-level simulators that are needed for the evaluation of battle management architectures and strategies under a large set of attack scenarios.

The adequacy of the battle management architecture and strategies will be heavily influenced by the thoroughness of the attack tests. The attack scenarios should be implemented in whatever level of detail that is needed to model anticipated enemy actions.

An important task of the simulation is to interject various errors into the battle management system and to simulate failures and loss of components in order to assess the system's ability to cope with such problems. NASA's experience in using "trouble-simulators" has proved itself most valuable.

In general, throughout all the software development SDIO should insist that each software delivery be accompanied by a plan for testing to verify that it meets its specifications, such that it can be retested whenever changes are introduced to it or to its environment, whether they are expected to influence it or not. The test plan should cover both the pre- and the post-deployment ("in-line") testing, and the ways to verify that the actual implementation match the prototype simulation.

B. Recommendations

The panel recommends that SDIO:

- Construct several distributed and diverse but centrally-coordinated simulators at several levels for testing and debugging the strategic defense system and evaluating BM/C^3 strategies. Make the simulators available for use over computer communication networks.

- Support general and specific research in simulation verification and validation.

- Develop simulations of weapon, sensor, and communication systems through joint efforts with the developers of these systems, and validate the abstractions of the various system components.

- Implement early high-level simulators that permit the evaluation of battle management architectures and strategies under a wide variety of attack scenarios, and couple them early to the system architecture and battle management work.

C. Summary

The strategic defense system requires simulators at several levels: for architectural studies, for battle management evaluation, for software development and method testing, and for in-line testing. Interoperability must be maintained among the various simulators.

Activities at the software simulation facilities will often be at the state of the art. Both fundamental research and development activities should be pursued.

For the architectural simulation studies, it is very important that SDIO devote substantial resources to "red teams" that devise attack scenarios. The panel appreciates that SDIO has already decided to do that.

V. SOFTWARE

The panel has many recommendations of software-related research and development for SDIO to support. We describe first software efforts that are specifically related to SDI. These are exploratory development projects that the panel recommends for examining the feasibility of proposed architectures for the battle management software. The second section lists basic research topics. Advances in these areas would have an impact on strategic defense as well as on other software applications. They are particularly directed at improving the productivity of the developers of a battle management system and the quality of the end product. The third and final section is a discussion of the nature of software research. It is important that those who will manage software research or make decisions based on its results understand its limitations.

This chapter proposes a framework in which to conduct research aimed at a better understanding of the battle management software and the technology that would support its construction. It does not contain intricate technical discussions about the appropriate modular structure for various key parts of the battle management software. It is not a design document.

The contents of this chapter are derived from the panel's discussions of the following basic questions:

- What possible battle management system architectures are the most feasible candidates for successful, dependable implementation?
  - How can one assess the likelihood of a candidate architecture for successful, dependable implementation?
- What conceivable advances in software research would make a difference in a battle management system's implementation?
  - How does one evaluate whether these advances will make a difference?

The panel has concluded that feasible system architectures exist. As discussed in Chapter III (Architecture), it is possible to build a trustworthy strategic defense system based on:

- Choosing simple, even if suboptimal, designs, since their behavior is more predictable over a wide variety of circumstances.

- Using highly decentralized designs so that testing of a relatively small part of the system would provide evidence from which to infer the performance of the entire system.
- Using, at several levels of the system design, a diversity of design approaches and implementations, so that an error or other vulnerability in one component is not repeated in all.

The panel advocates an *open system* approach to the design of the battle management software. Open systems are characterized by a concurrent, message-oriented programming model and narrow semantic interfaces between modules. Existing open systems have several properties that would be desirable for the battle management software: It is easy to add or replace software modules. Often they are robust under a wide range of possible subsystem failures.

A. Discussion

1. Exploratory Development

The Brooks Law of Prototyping†:
"Plan to throw one away; you will, anyhow".

The key software-related question is how to assess the feasibility of the software for a given battle management system architecture. There are no analytical methods that can answer that question. The only approach available is to propose a range of possible system architectures and assess, as systematically as possible, their effectiveness and dependability.

To this end, the panel recommends that SDIO initiate the construction of several prototype battle management software systems. These exploratory development efforts are intended to probe the battle management software design space. Ideally, a quite diverse set of design approaches will be employed, including a variety of open system designs.

SDIO's selection criteria for these projects, and furthermore its evaluation criteria, to be repeatedly applied in increasing detail throughout the lifetimes of the

† Frederick P. Brooks, Jr., *The Mythical Man-Month: Essays on Software Engineering*, Addison-Wesley, 1974. In the paragraph introducing this piece of time-honored advice: "The management question, therefore, is not *whether* to build a pilot system and throw it away. You *will* do that. The only question is whether to plan in advance to build a throwaway, or to promise to deliver the throwaway to customers."

projects, should be those criteria defined in Chapter III (Architecture): performance, cost, and testability. SDIO's means for increasingly detailed evaluation should be the simulation system described in Chapter IV (Simulation). It is a challenging requirement on the simulation system that it be capable of simulating both the execution of these prototypes and the scenarios against which they are tested, and at increasing levels of detail.

These battle management prototyping projects should start out relatively small, at about the 25 man-year level. Initially, the projects could work with abstractions or simulated approximations of the sensor and weapon characteristics. Each project should be capable of being expanded to a much larger scale exploratory development. As the prototype battle management systems increase in size and capability, the level of realism and detail in the simulations would be increased.

Since the strategic defense system will be distributed among many assets, the battle management prototypes should, from their very beginning, be designed for operation over a computer communication network.

If more than one prototype battle management system were developed into a system ready for deployment, all the better. This outcome would be consistent with the panel's philosophy of diversity and decentralization. Still, the ability ultimately to expand a prototype to the development of a deployable system should be treated as a goal but not as an inevitable outcome. This goal prevents ignoring problems that a serious development might have to face. Furthermore, this goal provides a criterion for measuring the progress of each project: how close they got to the finish line (as opposed to how far they got from the start).

Contractors should be free to choose their own development environments, methods, and design approaches. They should be encouraged to experiment with new tools and techniques and with unorthodox structuring of the development organization, except for the requirement that such experiments be instrumented as carefully as possible.

It appears that the misconceptions of some software researchers complement the factual knowledge of software development practitioners, and vice versa. One misconception that each shares, however, is a belief in the reality of the "waterfall diagram", the portrayal of a software development project proceeding smoothly and unidirectionally from requirements specification to design specification to coding, testing, and delivery. The truth is that these steps can, and often do, feed back into any of their predecessors. In particular, the basic adequacy of a system design is determined through a wide variety of formal and informal feedback mechanisms: interactions between customer and implementer, design walkthroughs, independent validation teams, simulation, testing, and operational experience. Accordingly, SDIO should consciously attempt to exploit the potential informative power of these feedback mechanisms while exploring possible software system architectures. SDIO should be as prepared to start new projects at "square one" as it is to expand existing ones. This approach allows the insights obtained previously to be taken into account. An already-contracted project might even be allowed to start over. SDIO needs to encourage feedback of the form, "Our first approach didn't work", or "The problem you asked me to solve turned out to be ill-posed".

SDIO should structure this prototyping program around two major goals: the assessment of the feasibility of a range of possible battle management system architectures and the establishment of the specifications of the system. There is a tradeoff between the effectiveness of a strategic defense and the complexity and interdependence of its software. Software that tightly coordinates the actions of every platform to insure optimal use of the weapons would be relatively fragile and difficult to test. If the system can not be tested convincingly, the defense's effectiveness will never be any other than hypothetical. Simulations can provide approximate measures of the effectiveness of prototype software architectures; at the same time their testability and robustness under many different conditions should be estimated.

Complete and precise specifications of the requirements are vital to the development of software. The prototypes are attempts to state the specifications of this system. This is the concept that is behind the emphasis on iteration and feedback in the prototyping program: If the specification (prototype system) is found to be erroneous or incomplete, it can be revised.

The panel sees this set of projects as an experimental, exploratory effort to better determine the most feasible battle management system design approaches. Therefore, the procurement should be treated more like a research contract than as a development contract. The government does not know exactly what it wants; research will determine what it needs.

These projects may expect to produce as deliverables both the prototype battle management software and precise definitions of the formulation of the system in terms of modules and the interface protocols among them. Unbiased critiques of each design approach should also be available to the government. Because of the unconventional character of the deliverables, contracting of these projects must be executed by creative, flexible, and somewhat technically sophisticated people. Not everyone involved in DoD procurement fits that description.

2. Basic Research Topics

The previous section is the "development" part of the panel's recommendations for software research and development in support of the battle management system. What follows is a list of research topics selected on the basis of their chances of improving strategic defense software productivity and the quality of the end product.

This list should not be interpreted as exhaustive or all-inclusive.

The distinction between research and development is, however, difficult to make. There are several examples from computer science history where clever tricks invented by system developers in the course of a development project were the inspiration for subsequent fundamental insights into that type of design problem. A better understanding of the battle management system's basic structuring might emerge from attempts to construct prototypes, just as experiences from early compiler-writing projects fed back into the academic community and led to such a clear understanding of compilation that modern compilers are, generally speaking, all structured in the same way. It will be important that results be shared between the prototype system developers and software researchers investigating the following topics.

a. *Massive Computing Power for Software Development*

The panel recommends that SDIO support research on how software development productivity would benefit from the availability of massive computing power in the development environment.

The use of supercomputers – high speed computers that may or may not resemble the current high speed scientific computers – may allow software developers to create new tools and techniques. For example, it may be possible to build much higher quality debugging environments or to support much more detailed semantic checkers than are currently available. Another interesting possibility is to exploit real-time animation of algorithms. Researchers have already demonstrated that such graphic support of algorithm development is a very powerful technique. The panel expects that these and other innovations may be possible if supercomputers are available to software developers.

The panel recommends that this research program begin with many small studies. These studies should determine in finer detail the possible uses and benefits of supercomputers for software development. If these studies produce promising results, then the use of supercomputers for would be justified for strategic defense software development projects.

b. *Software Testing*

The art of software testing amounts to a search for good compromises. The range of possible inputs to any useful program is too large for the program to be tested exhaustively. Innovative approaches might be found for selecting test sets in such a way that they "cover" large sets of possible inputs and all possible behaviors ("branches"). Ways to exploit high computing speed in the testing environment would be of interest. In particular, the panel recommends the investigation of testing methods and tools that could be an integral part of a software development environment, enabling a system to be tested at all levels of its structural hierarchy and throughout the development process.

c. Software for Reliable Systems

Software engineering researchers have perhaps been overly concerned with finding rigorous mathematical approaches to the design and analysis of software. These approaches are intended to assure that the software is flawless. The panel recommends that SDIO place considerable emphasis on the invention, refinement, or evaluation of software design techniques that would allow computing functions to be performed usefully despite hardware faults or software errors.

Writing software for computers designed to tolerate hardware faults is currently a complex and difficult task. A combination of new architectures and new programming techniques might make it easier to program fault-tolerant hardware.

A few approaches to developing software that can override or compensate for errors have been proposed. One example is "N-version" programming, or "design diversity", in which several programming teams write a program to the same specification. The programs are run together, with some decision procedure selecting the "correct answer" from among their outputs. Larger experiments than have previously been feasible would contribute to the evaluation of this approach. Examinations of the likelihood of highly correlated programmer errors should also be undertaken, since such a phenomenon would argue against the use of the technique.

Other approaches entail techniques or paradigms in which a software module attempts to detect, correct, or protect itself from errors occurring in different modules. Watchdog processes, data structure audits, and recovery blocks fall into this category. Two additional run-time techniques are discussed below.

All such software error-masking approaches are computationally demanding. It is important to understand the hardware and real-time tradeoffs involved in their use.

c.1 Runtime Checking

It may be advantageous to employ more extensive runtime checking of program execution in a battle management system than is used in common computing practice, even if it requires abundant computing power. Such runtime checking could detect and sometimes correct programming errors. If research shows that such techniques are useful for battle management computations, they could be used selectively in a deployed system. Critical computations such as weapons release could be implemented in this "ultraconservative" scheme, while noncritical computations such as an antenna-pointing schedule could be implemented conventionally.

A simple example of a useful runtime test is to check bounds on values assigned to variables. Bounds checking is commonly used for array indices; one could also place bounds on the values of physical quantities to enforce a "sanity check" on the results of a computation. Another example of runtime checking is to tag numerical values with their physical units. An attempt to perform an addition of quantities tagged with different physical units – such as meters and grams – would be detected and flagged as a fatal error. An attempt to add quantities expressed in meters to a quantity expressed in nautical miles would result in an error being flagged and a conversion automatically performed. These examples are a simple extrapolation of a type of runtime checking that is already performed by many programming systems.

c.2 Probabilistic Techniques

Decisions based on probability distributions rather than on a single item of information provide a control on the extent to which single faults or programming errors could otherwise propagate to corrupt the reliability and performance of a battle management system. As counterintuitive as it may seem that the explicit use of probabilistic measures could increase the reliability of a battle management system, the panel suggests that research in such disciplines may well prove to be fruitful..

For example, critical data items could be represented not just as single numbers, but with associated probability density functions. These probability density functions could be represented very simply with "error bars" on data, or by more precise functional representations if the reliability of the data could be estimated in this way.

In combining data from several sensors, one could weight the data from the different sensors according to their reliability. A single specific number representing launches detected – for instance, 600 – is much less valuable than a count represented as 600±400. Such a large error might result if there were discrepancies between multiple sensors, and would cast doubt on the threat assessment. On the other hand, a count of 600±4, which could occur only if there were substantial agreement in several independent assessments, could provide a high confidence level for a decision to activate the defense.

At a higher level, modules should assess the credibility of each other, and share this credibility measure with other modules. This sharing may help the system as a whole to detect malfunction of some components, and take a corrective action such as instructing the malfunctioning unit to "reboot" itself, or suggesting to other units to ignore the malfunctioning one.

c.3 Verification and Other Mathematical Proof Techniques

The panel does not recommend abandoning formal techniques. To the contrary, the panel encourages SDIO to apply them as appropriate, but wishes to caution SDIO on the limitations of their use. Mathematical proof techniques successfully applied to certain very critical modules or protocols would increase confidence in programs. The codes concerned with weapons release and with communication security are candidates for such verification, since they are obviously critical and possibly meet the other criteria as well. In keeping with the main theme of this report, the panel suggests that the ability to verify these important modules is not as vital to establishing their trustworthiness as is defining them, from the start, to be as simple and straightforward in function as possible.

d. Specification Systems

A clear, precise specification of a software design is crucial. Several different notations for specifying software exist. A few are commercial products; others are in the research stage. The panel recommends research support for that class of specification languages intended to serve as more than just an independent, documentary description of the software design.

One class of specification languages is those designed so that programs written in them can be iteratively refined. As successive levels of detail are added, the programmer has strong assurance that the semantics of the more detailed representation match those of the previous iteration. Another class - one that has a large intersection with the first - is composed of specification languages designed to be executable. An executable specification language offers several advantages. It forces the specifier to be precise and to avoid ambiguity. It ties in closely with concepts of simulation and testing at every level of refinement. It would be an asset to the iterative, exploratory style we prescribe for the battle management software development projects discussed previously. Research prototypes in this class exist, but are presently limited in usefulness. They are often slow to execute, their abstruse mathematical notations make them difficult to use, and they are unable to express some types of software requirements, such as real-time constraints.

e. *Parallel, Concurrent, and Distributed Computing*†

Parallel computing is a necessary direction for SDIO to take in the realm of algorithmically specialized processor arrays for signal and low-level image processing.

The question is open, however, as to the extent to which more general-purpose, programmable concurrent machines could be of use for the rapid execution of higher-level, less regularly structured battle management functions. Small- to medium-scale concurrent computers have recently become commercially available, and several large-scale experimental prototypes are being developed. However, the ability to exploit their speed over a broad range of applications lags behind.

SDIO should establish centers for studying algorithmic, compilation, and operating systems issues in the context of programmable concurrent computer systems with the potential of having fairly general application domains. Multiprocessors located at these centers should be available to SDIO contractors for experimentation via remote access, as described in Chapter VII (Communication).

f. *Organizational Issues*

The advent of networks of computer workstations, precise specification languages, and other components of advanced software development, creates the possibility of structuring software development teams in entirely new ways. Computer networks can support distributed development of software, unlike the more conventional geographically colocated efforts. A "cottage industry" of small, informally organized groups could conceivably make software changes easier and faster while avoiding mistakes. One or two carefully designed experiments with radically structured software development teams might be a good choice. It might be worthwhile to conduct a historical study of a large military software system development project to assess the technical, managerial, and organizational aspects of its success or failure.

g. *Exploratory Programming*

SDIO might take a look at which environments enable programmers to be most productive. Programmers with access to high-speed, tool-rich environments have created a style of program development that seems to be based on an attitude that such environments make programming at all levels to be less labor-intensive. Programmers seem to feel free to do things that in previous environments would

† These terms are often used interchangeably. To the extent that we make a distinction, we use *parallel* to suggest "lockstep" actions, as in a "parallel adder" or SIMD computer; *concurrent* to suggest partial independence of control, as in an MIMD computer, and *distributed* to suggest that the distribution of control and data represents a substantial cost in comparison with computation.

be prohibitively expensive, such as tearing a program apart and putting it back together in a new structure. It might be worthwhile to determine and study the factors that support rapid prototyping and this exploratory style of programming. The goal of the studies would be to determine how the programming style and the powerful environments can best be exploited.

h. *Software Maintenance and Reuse*

"Maintenance" refers to the continuing process of changing a software system while it is in service. Belated discovery of errors and omissions dictate modifying the code of a delivered system. Maintenance accounts for a significant fraction of the overall cost of a software system. A standard rule of thumb is to estimate the maintenance cost share at 70 percent. A deployed battle management system would undoubtedly require a substantial software maintenance effort. The panel expects that the system would be continually tested before, during, and after deployment, and that this testing would expose system bugs and inadequacies.

Introduction of new technologies into the defense system and responses to newly discovered countermeasures or threats will require updates to software that is beyond what is implied by the term "maintenance". New versions of a battle management system, involving the reuse of substantial portions of previous versions, will be produced at regular intervals.

It is as important to apply the same rigorous discipline to the design, verification, and testing of new versions of a system as it is to the original implementation.

Both maintenance and reuse are simplified by a careful choice of the way in which the system is partitioned into modules. However, there has been little research on tools for modifying software systems. The panel proposes that attention be given to the question of what methods and tools might be applied not only at design time, but throughout the development process. Methods might be developed to determine and monitor the modules of code that reflect each system requirement, easing the job of accommodating a change to that requirement even if the affected code is dispersed rather than isolated.

3. On the Nature and Limitations of Software Research

a. *Software research can not be expected to make radical advances*

The panel is certain that no technological means will be as vital to the successful development of the battle management software than the choice of the right end. SDIO must not undertake an arbitrarily difficult software development task, expecting a breakthrough in software research to happen in time to save the day. Finding a feasible system architecture is possible, but making radical changes in the way software is developed is not. The overall system architecture must be designed around a predictable, reliable battle management system that could be implemented and tested using existing software development technology. Any other choice puts a fundamental research problem with no promising line of investigation in the critical path.

To be specific, the panel expects no technological breakthrough that would make it possible to write the celebrated "10 million lines of error-free code". To base the successful deployment of a strategic defense system on such a breakthrough would be unrealistic. It will be important to develop highly reliable software for the battle management system, and advanced software engineering tools will surely help, but they will only help reduce the number of errors, not eliminate them. The panel recommends, instead, the pursuit of approaches that allow the system to function dependably despite the presence of some errors. These approaches should be reflected in the overall system architecture as well as in the software research SDIO supports.

Other computing technologies have more "elasticity" than software. For example, the computer science community reasonably expects computing hardware speeds to increase significantly. SDIO should focus attention on exploiting advances that are likely to occur and on exploiting techniques in which we are already proficient. There are types of software that we understand how to build fairly well, such as compilers, data base management systems, and version control systems. Software development environments would certainly be enhanced by the availability of a rich set of these well-understood tools and significantly faster hardware. Those enhancements can not, however, be expected to make the impossible possible.

One additional note of warning on a related subject: We find it a bit troublesome to be discussing whether radical advancements in software technology would enhance the quality of a new defense system, when we are aware that many of the DoD's biggest software development contractors are presently literally decades behind the state of the art – an art that is only a few decades old. Suppose new technologies do come into being. Are we sure they will be used?

b. Software Research Can Not Predict Much

Computer Science is a much younger discipline than physics, whose constraints are much more thoroughly understood. No one suggests, for example, solving the difficult power-generation problems peculiar to deploying beam weapons in space by orbiting perpetual-motion machines. On the other hand, the limits of effective computability, although well understood, do not place strong constraints on what is possible. In particular, there are no laws or formulæ that can accurately predict the success or failure of a large software development. Sooner or later, SDIO will have to judge which proposed battle management system architecture is most feasible. This decision will have to be, at least in part, qualitative.

Predictions of the feasibility of a software system design are fairly accurate only if the predictor has experience implementing a quite similar design. This consideration motivates the panel's recommendation of the exploratory prototype development projects. Software developers could gain insights into the best ways to partition the system into modules; some critical problems could surface. Prototyping does not, however, provide complete understanding.

Just as in many other fields of engineering, a system that is much larger than its earlier prototype may cause an entirely new set of development problems to become dominant. Even with the extensive simulations of the prototypes we assume will be performed, the information obtained from a prototyping project about the feasibility of a deployable version will be approximate.

The inability of software experts to back up their judgements of the feasibility of implementing highly complex systems with "hard figures" places them at a disadvantage in establishing optimal tradeoffs between system complexity and hardware. Qualitative objections to some software-intensive ploy to save hardware costs might be overwhelmed because "hard" dollar figures can always be trotted out to support the "tricky" approaches. SDIO should adopt a policy now, despite the absence of a "one-line proof" of its necessity, of consistently opting for the simplest, cleanest software design possible.

c. *Software Research Can Not Evaluate Much*

Given that SDIO will conduct research in advanced techniques and tools for developing software, it would be appropriate to have a way of measuring their quality and their applicability to the development of a battle management system. No such measurement is possible today. The basic reason is that software is not, in the classical sense, an experimental discipline.

This is not to say that software engineering is not an experimental discipline; it is indeed such. It is that software experiments are not designed around testable hypotheses, as in the classical experimentalist paradigm. A testable hypothesis is one that can be proved *or disproved* by the result of a properly designed experiment.

Usually, what software researchers call experimentation is the construction of prototypes. There is rarely a hypothesis being tested that is any more general than "It is possible (for me and my students) to build a small system with property x using technique y". This constructive, "existence proof" method of investigation is prevalent in other subfields of computer science as well, and is probably adequate, whenever the primary concern is with design issues.

In software research, however, aspects of human behavior must be taken into account. Issues of how humans can comprehend notations, express computations in them, and exchange complex, detailed information with one another are paramount.

The lack of empirical observations of human behavior in an experimental setting therefore limits our ability to measure the effectiveness we seek.

It is not due to ignorance or negligence that software research rarely employs the classical experimental paradigm. It is simply that meaningful experiments are practically impossible to construct. Large software development projects require many programmers over long periods of time and, therefore, are expensive. Projects small enough to be affordable, on the other hand, do not necessarily provide insights applicable to larger projects. Furthermore, software development experiments are practically impossible to control: There are too many variables. If a programmer is unsuccessful at using a new programming language to implement the module he is assigned, is it due to an inherent inadequacy in the language, the language's inappropriateness for that type of application, the programmer's inexperience with the language, or his inability or unwillingness to learn it? A large set of experimental projects might allow such local variations to be statistically filtered out, but they would be even more expensive. Thus, assessments of software development techniques have been largely qualitative. Indeed, many of the well-known papers in the programming languages and operating systems disciplines have a distinct flavor of literary criticism.

The conclusion is simple: for most of the software research issues that have a phenomenological component, the discipline is unequipped to supply indisputable proofs or precisely measurable results. This differentiates software from most of the other scientific aspects of the Strategic Defense Initiative. For example, if one wants to decide whether free electron lasers are viable candidates for deployment as antimissile weapons, one can probably construct a logically chained sequence of experiments based on testable hypotheses. If properly designed and conducted, the experiments would settle the issue quantitatively and conclusively. On the other hand, if one wants to decide which software development technique is most appropriate for a particular subset of the battle management software, one can not make an objective assessment; it will rely at least partially on anecdotal evidence and the subjective judgment of experienced people.

B. Recommendations

The panel recommends that SDIO:

- Support a few prototyping projects for the development of battle management systems, with emphasis on the methods used, such that these efforts could be developed into full scale deployable systems. These prototypes must exhibit the properties important for the strategic defense system, such as responsiveness, robustness, network based, testability, and design for diversity and for evolution.

- Manage the above battle management prototyping and the architecture projects such that they can influence each other.

- Support research in software technology in the directions that are expected to influence and improve the development of the software for the strategic defense system. Section A.2 of this chapter has a list of such topics. SDIO should manage these projects to assure their influence on the software development efforts.

- SDIO should *not* select a particular software method (or style) as an exclusive method for strategic defense software development.

C. Summary

The feasibility of the battle management software and the ability to test, simulate, and modify the system are very sensitive to the choice of system architecture. In particular, the feasibility of the battle management system software is much more sensitive to the system architecture than it is to the choice of software engineering technique.

Software technology is developing against what appears today to be relatively inflexible limits in the complexity of systems. The tradeoffs necessary to make the software tractable are in the system architecture.

The panel does not recommend that SDIO choose a particular software development method and declare it to be the preferred or exclusive approach. Instead the panel recommends that SDIO support several diverse battle management system prototyping projects that will illuminate the specific software issues, and would contribute to the specification of the deployable software.

In addition, the panel recommends that SDIO sponsor research in software technology along directions that have potential to benefit the battle management system development.

VI. HARDWARE

Unlike software technology, where there appears to be little growth potential, the panel's expectation for hardware technology is one of continued rapid progress. Advances both in devices and in architectures are leading to systems in which increased computing speed is combined with reduction in size, weight, and power. This progress is a result of ongoing commercial and government-sponsored efforts such as the Very High Speed Integrated Circuits (VHSIC) program of DoD and the VLSI and the Strategic Computing programs of DARPA.

This report, unlike the Fletcher Report, implicitly assumes that spaceborne computing power will be plentiful. It is important to make this assumption a reality.

SDIO should exploit advanced computer technologies to simplify the more difficult tasks. The architects and system implementers should ask themselves how they can use high-speed computation to simplify a task, rather than how a task can be performed with minimal computing capabilities. Initial review of Phase I architectures revealed a very conservative proposed use of computers in all areas.

The panel recommends extensive use of computers in all activities related to SDI. Some actions are needed to prepare the hardware technology before it can serve these needs. This chapter deals with specific actions necessary to better exploit the hardware potential. The panel recommends investments that will advance hardware technology in time for deployment decisions.

A. Discussion

SDI will employ digital computing hardware in these areas:

- *Object Computers:* These are space- and ground-based computers that perform the computing functions required for battle management, signal processing, and communication. Although signal and image processing may be considered outside the scope of the SDI BM/C^3, the computer requirements for SDI must not be addressed separately for BM/C^3 and surveillance, acquisition, tracking, and kill assessment (SATKA). In the past, designers tried to be conservative about hardware and made the software overly complex to compensate for the lack of computer power. The panel recommends use of massive computer power to simplify other tasks.

- *Simulation Computers:* These are ground-based systems used to simulate the battle management system at various levels of abstraction, from the high conceptual-level simulation to the lower-level detailed simulation and testing. At the low testing level, the simulators could be coupled with the deployed system to perform *in-line* system tests. Early concept-level simulation of the battle management system may run on a single computer, but all other levels will require massive amounts of computation that might be available only from highly concurrent multiprocessor systems and supercomputers. Simulation engines capable of executing thousands of scenarios are necessary. These simulations must operate quickly to allow the designers, experimenters, and testers to perform their tasks efficiently. The simulators will be required to run faster than the machines they simulate. It will be desirable and feasible to deliver to each simulation center computing engines that perform simulations significantly faster than current computers.

- *Software Development Computers:* These are the computers used for the development of the battle management software. The panel encourages use of the best available computing facilities in this area. In Chapter V (Software) of this report, the panel strongly recommends significantly increased use of computing resources to increase the effectiveness of software development. Because different groups will use different development systems, the panel recommends that their facilities be interconnected (via computer communication networks) to encourage and facilitate sharing. The current procurement practice that discourages capital investment has greatly hampered innovation in defense system development. This situation is unacceptable for SDI software development activities, which could benefit from increased availability of computing resources.

- *Hardware Development Facilities:* These are the facilities that hardware developers use to increase their effectiveness. These facilities include powerful integrated tools that allow design and verification of hardware, which, when fabricated, will quickly qualify for use in SDI/space applications. Recently, the design and simulation of very large scale integrated circuits (VLSI) has forced the industry to use larger amounts of computational power. Experience shows that massive use of computation in the verification and validation of custom chip designs significantly decreases the time required to obtain working chips. Again, discouragement of capital investment through competitive procurement policies has hindered defense system designers. In some industrial organizations innovative VLSI designers use significantly more advanced computers. more sophisticated tools, and more computing cycles than their counterparts designing for the U.S. government. Hardware developers should be linked via communications networks to encourage sharing of information, of design tools, and of the expensive resources needed to support design and verification.

1. Technology Insertion Delays

The United States has an advantage over the Soviet Union in hardware technology and in the infrastructure that supports hardware fabrication. We lose some of our advantage in space applications because of delays in inserting new technologies. We must reverse this trend. We must work to better understand the reasons for these delays and innovate faster methods for technology insertion. There are several primary reasons for the delay in technology insertion for space applications:

- The long delay between the development of a new technology and its availability to end users.
- The sequential nature of accelerated life (*step stress*) testing for qualification.
- The serial order of development of a dependent technology after a supporting technology has fully matured.
- The lack of general testing and space-qualification methods, especially for circuits with large numbers of internal states.

In many cases new technologies have been more reliable and affordable but the momentum of the qualification procedures has favored older technologies. The VHSIC project had to take great initiative to accelerate the use of newer VLSI technologies.

Better understanding of the space environment and the behavior of hardware in space is now available. The current practice of conservatism in use of technologies stems from an overly restrictive qualification process that is based on lack of proper appreciation for the behavior of hardware under space environments, and a time-consuming, unautomated inspection and tracking procedure.

An infrastructure to allow architects and designers to confidently use the current technologies should be developed and regularly upgraded with new knowledge, tools, and facilities.

SDIO should develop special design rules such that parts designed according to them will be easier to qualify for SDI/space applications. This knowledge can be coded in software tools and then used by hardware designers. Hardware designed using these tools will qualify rapidly for SDI/space applications. The industry is already experienced in designing VLSI circuits with software tools. The intricate design rules (based on the observed behavior of devices built according to identical processes) are coded in the software tools. Circuits designed using these tools have typically performed correctly on "first silicon". Availability of an infrastructure integrating such tools will help the innovative designers implement their designs directly in the SDI/space-qualified technology. The most creative designers will make a variety of technical tradeoffs to accommodate the specific constraints. The current practice of reengineering earlier designs (optimized for unrelated design constraints) without creative designers is not acceptable for the SDI application. Innovative designers, who are rare, must find it interesting to design for SDI.

To solve the problem of the lag in technology insertion, it will be necessary to propagate a different culture of system development that will exploit the emerging technologies. Insertion of technologies requires a continuous upgrading of methods consistent with new techniques. The endless demands of project schedules, the lack of capable staff, the lack of capital equipment, the "not-invented-here" syndrome, the conservatism in procurement decisions, and bureaucracy have created a culture that resists change and takes only naive risks. SDIO must create a new culture that can adapt to changes more effectively.

As part of such a culture, new technologies and their design methods would be separately certified and qualified for space/SDI applications. After that, standard cells and other functional building blocks should be designed, qualified, and made available to the designers. The design methods and the standard cells should support the testing requirements of space/SDI.

In addition, new approaches to space/SDI qualification should be developed. This includes test structures for ongoing monitoring of environmental effects such as radiation and other wearout mechanisms.

For other hardware areas such as microprocessors, several supporting activities in progress are encouraged by DARPA, the VHSIC program office, the commercial users, and others. The panel prefers building on the results of other research and industrial development rather than duplicating efforts. Parallel efforts in related hardware technologies must be pursued with coordination and periodic crossover.

The experience gained among workers in the DARPA-sponsored projects has clearly demonstrated the value of sharing technical information and development tools. Encouraging reuse of technologies developed at other organizations is essential for rapid progress and best results per hardware research dollar.

SDIO should accelerate the insertion of new technologies to the space environment, by creating the needed infrastructure, and by taking steps to accelerate the space/SDI qualification process.

The dominant cost items in the estimates presented to the panel regarding the price of a deployed strategic defense system are the "hardware" of the defense system (such as satellites, weapons, and sensors), and placing that hardware in orbit.

Due to the large manufacturing volume of the SDI system, SDIO must support development of automated manufacturing aimed at lowering the manufacturing cost of the space defense hardware by "mass production". Automated manufacturing is a multidisciplinary issue involving robotics, materials science, and manufacturing science. SDIO should seek advice in this area from the center of excellence in automated manufacturing research at the National Bureau of Standards (NBS).

In fact, SDIO should examine a *Jeep* approach in addition to the traditional and dominant *Rolls-Royce* approach to space assets.

2. Computing Speeds

Computing speeds have been improved already, and further improvement is expected independent of SDI. However, advancement in algorithmically specialized processors and in concurrent computer architectures has been very slow because of the research nature of this field and because of limited interest in extremely small size and power. Until now, size and power have been reduced with the help of the smaller geometry of VLSI rather than of newer architectures. Current designs use functions such as arithmetic operations to build larger functions. Researchers must start using higher-level functions as primitives to build advanced systems with much greater capabilities.

SDIO should develop higher-level reusable functions that are fully optimized for the highest performance while using the least amount of power and space.

Signal and image processing systems are currently designed with computing functions such as arithmetic operators. Often, these functions are simplified by using fixed-point arithmetic to minimize the hardware task at the expense of considerable system and software complexity. Higher-level functions must be optimized and standardized. A factor of 10,000 improvement in size, power, and weight through the use of algorithmically specialized processors such as systolic arrays and Fast Fourier Transform (FFT) processors has been demonstrated. It is also important to identify and develop specialized, non-numeric functions suitable for use in SDI. Specialized processors, instead of lower-level primitives, will then be used to build systems.

Use of algorithmically specialized processors demands the development of heterogeneous computer architectures. These new architectures will be designed to cope with the massive computing capabilities of these new components. Some work in this area has been sponsored by DARPA's Strategic Computing program and by ONR.

To take advantage of the emerging parallel processor architectures, it will be necessary to develop components, interfaces, and control structures. These developments will be influenced by the packaging technology. Therefore, very dense packaging for lowest power and weight plus the fastest speeds must be available to the designers and architects. It is a common practice among commercial computer manufacturers to develop packaging technology in parallel with the architecture.

SDIO should develop packaging technologies for SDI and set goals sufficiently in advance to allow architectural innovations.

3. Reliability and Availability

The special nature of SDI dictates that computing assets will have to be operational for a very long time without manual intervention. Due to the limited interest in this issue, significant computers that can operate unattended for several years are not available and are treated as an exception. For comparison, the Shuttle computers are designed with MTBF of 1,000 hours (about a month), which is two orders of magnitude less than what is needed for satellites that are expected to be operational for 10 years. Therefore, the panel recommends that SDIO direct efforts in the direction of achieving full-scale computers with hardware and software capable of operating without manual intervention for periods such as 10 years.

SDIO should invest in hardware technologies to substantially increase computing speeds at the lowest cost in power and weight. Algorithmically specialized processors and fault-tolerant hardware are some of the most promising technologies. The need for long operational life in space must also be addressed.

Faults will be caused both by random hardware and software failures and by deliberate hostile actions. When failures occur, and recovery is not possible, the computing capabilities must degrade gracefully and not collapse completely. Distributed computing with loosely coupled sets of tightly coupled multiprocessors is an emerging technology that offers graceful degradation under fault. It is now possible to install excess capacity that will tolerate loss of some computing elements. The SDI computer architectures must utilize this technology. Miniaturization has significantly increased hardware reliability and availability.

Further improvement is expected in this area. Traditionally, commercial semiconductor manufacturing facilities have been able to process large wafers with very low defect density. In mass-produced commercial circuits high yield is obtained through use of automation and highly stable foundry operation.

Physical presence and involvement of human operators and unstable processing conditions cause low yield and erratic results from unautomated research foundries that have frequent changes in the processing parameters. The VHSIC contractors suffer from low yield because of lack of automation and lack of stable foundry operation.

SDIO should examine the creation of a highly automated SDI foundry (as part of the infrastructure) that will produce Space/SDI qualified parts with high yield and quick turnaround.

SDIO also needs to develop technologies to cope with nuclear effects on space assets. The electromagnetic pulse (EMP) phenomenon is readily dealt with by electrical shielding. However, the high-energy neutron flux from a nuclear explosion is expected to "erase" volatile semiconductor memory from a substantial distance ($\approx 10^3$ kilometers). Effective shielding is difficult. Transient communication and computation upsets of several microseconds may occur frequently in any orbital system. A conservative system design must commit a large volume of data and programs to nonvolatile secondary storage systems. SDIO needs to develop and qualify the technologies for secondary storage systems that would be immune to high-energy particle upsets. Automatic system restarts from secondary storage may be required frequently, so restarts must be fast. Certain vital information (*e.g.*, time, orbital elements for restoring communication, and parts of the task queue) will have to be stored in a way that is immune from these upsets.

The availability and reliability requirements of SDI/space applications necessitate extensive use of fault-tolerance technology. The methods of fault-tolerance at low functional levels using clever coding have been discussed in the technical literature. The VHSIC Phase III projects further developed these concepts. Use of these concepts has been limited to memory designs and to protocols implemented through software, as in the DARPA/RADC Advanced Onboard Signal Processor (AOSP) project. The SDI demand for reliable operation can be well served by upgrading the current practice with early demonstrations of advanced, fault-tolerant hardware technology.

Hardware fault-tolerance to natural hardware failures makes it difficult to program the computers unless adequate consideration is given to the programming aspects of fault-tolerant architectures. The fault-tolerance of memory systems is a good example of a design that frees the software developer from the inconvenience of explicit actions. Traditionally, recovery from a hardware failure (transient errors in particular) has been very difficult. Typically, recovery methods are superimposed on the system after it is designed, which causes time-consuming, rough methods of recovery from failures in data processing systems. Interestingly, in SDI/space applications, the recovery process will not require a return to exactly the pre-failure state.

SDIO should address the fault-tolerance and recovery issues specific to SDI and other space applications.

B. Recommendations

The panel recommends that research and exploratory development be pursued in the areas described previously. SDIO should discourage duplication of efforts under SDI in the areas well served by other initiatives. SDI participants, however, must be encouraged to take the initiative in using the results of those other efforts. The proposed infrastructure will help the participants confidently use the results of current research.

1. **Create Infrastructure for Rapid Technology Insertion**

- Study and define the real SDI/space qualification requirements and codify them into design rules for devices, packages, and systems.

- Study the reasons for delays in the qualification process and define new streamlined methods for SDI/space qualification, including design rule checkers and automated manufacturing tracking.

- Provide quick-turnaround fabrication facilities similar to the DARPA/MOSIS facility using processes and packages that are SDI/space qualified and are included in the design rule checks for SDI/space qualification.

- Provide an integrated hardware design tool set that is regularly updated with state-of-the-art tools from industry and universities. These tools should include checkers for the current SDI/space qualification design rules to guarantee qualified parts soon after fabrication.

- Develop a set of SDI-qualified standard cells and upgrade them as technology progresses.

- Develop SDI oriented packaging technology integrated with the fabrication technology discussed earlier. Availability of a dense and well-cooled package for use by all hardware designers is necessary. The packaging technology will include consideration for ease of future upgrades.

2. **Deliver Fastest Computing Speeds**

- Develop appropriate computer architectures. Architects should develop architectures for SDI requirements that deliver the fastest computing speeds while adhering to other constraints such as ease of enhancement and fault-tolerance. This should be a multiprocessor architecture that permits efficient use of heterogeneous processors, including algorithmically specialized processors.

- Develop algorithmically specialized processors. These algorithmically specialized processors are necessary to gain significant advantage in computing speeds. These modules will be reusable and will allow flexibility for use within a restricted domain without diluting the performance advantages. They will be designed to fit into the heterogeneous architecture described earlier.

- Develop an architecture optimized for SDI simulation applications.

3. Increase Availability and Reliability

- Develop technology for fast recovery from major upsets. This technology for fast recovery will address the question of retaining information in a robust part of the hardware that can withstand severe upsets. It will also address the question of how to regain operational capability in the face of loss of components.
- Develop methods for very long service life in space. These methods will address the advantages of error correction at low levels (*i.e.*, arithmetic error detection and correction) that is not a common practice at this time. Understanding of the causes and types of failures in the SDI/space environment must be integrated with these new designs from the beginning.

C. Summary

Research directed at hardware technology offers the potential for continued rapid progress. To support space applications, the panel recommends that SDIO sponsor research and development to create the infrastructure for rapid technology insertion, deliver the fastest computing speeds for SDI requirements, and increase hardware availability and reliability.

VII. COMMUNICATION

A strategic defense system requires secure, survivable, high-performance communications among all the spaceborne and ground-based assets.

Due to the specialized nature and spatial distribution of the strategic defense assets, several forms of coordination (*e.g.*, sensor data, advanced warnings, and handoffs) are required. The communication must be robust and able to cope with (innocent) failures, (malicious) losses, and a hostile nuclear environment (*e.g.*, EMP and neutron flux). It must also resist jamming and spoofing. The communication performance requirements are anticipated to be 1-10 Mbit/sec with less than a few tens of milliseconds delay among closely coupled (neighboring) assets and less than a second or two delay between any remote parties. The precise performance requirements depend highly on the system architecture; therefore, they are not well characterized at this time.

The requirement for high bandwidth implies the use of high frequencies limited to line-of-sight connectivity. The relative motion of the assets (relative to each other and to the ground) rearranges the physical grouping of the various platforms. Some of the communication assets may not be operational. The logical configuration of assets into battle groups must then occur dynamically. The motion causes the line-of-sight communication connectivity to change constantly. Therefore, the communication network must be able to rearrange its topology dynamically. This requirement implies the use of a communications network, because direct communications links will not always exist between every pair of assets that have to communicate (*e.g.*, between the boost phase detectors and the man-machine interface).

It is most likely that this network would be a store-and-forward type. However, in "local areas" (*e.g.*, within the line-of-sight domain of a battle group) there may be additional types of communication.

The communication network must have protection against enemy countermeasures such as jamming, spoofing, and destruction. Techniques for ECCM are required, as are methods of monitoring threats to the network (and its assets), assessing the threats, and responding appropriately (*e.g.*, retracting antennas). When the protection mechanisms fail, the network must exhibit graceful degradation of its performance and initiate automatic recovery procedures.

The entire communications network must also be protected from itself. For instance, it should operate in spite of problems such as its assets (nodes) transmitting excessive data and flooding the network, transmission of erroneous information (*e.g.*, routing), and equipment failure. In order to protect the network from itself, there will be a need for methods such as for guaranteeing loop-free and deadlock-free operation, load balancing, and graceful degradation under heavy traffic.

The algorithms for operating such a network require substantial research, development, and validation.

SDIO should pursue the necessary research and development to support this type of dynamically configured communication network.

A. Discussion

1. Dedicated Communication Network

Volume V of the Fletcher Report implies that the communication network be composed only of the strategic defense system's own assets. There are, however, several other possible architectures for the communication system. For example, the communication may be structured around a backbone, or subnet, consisting of many satellites dedicated solely, or mostly, to message processing and passing (*i.e.*, packet switching). This concept is similar to DARPA's proposed Multi-Satellite System (MSS). Such a scheme can support a variety of tradeoffs, such as between omnidirectionality and unidirectionality of transmission, power, and robustness. This dedicated communication backbone is a promising example of the alternative communication architectures that SDIO should explore.

The relative simplicity of satellites dedicated solely to communication results in low cost. Therefore, SDIO could afford to acquire enough satellites to achieve early dense coverage and redundancy. Such a dedicated communication network could be built and deployed as a few independent subnetworks (possibly by different vendors) that cooperate in a way that provides robustness and redundancy through diversity.

One of the main features of the separate communication system concept is achieving in early deployment a complete communication coverage to support the phased deployment of the more expensive strategic defense assets. The system provides coverage in an innocuous manner without violating existing strategic arms limitations treaties. Separating the communications system from more expensive assets allows communications improvements independent of modifications to the more expensive sensors and weapons.

Therefore, SDIO should study various communication architectures beyond what is implied by Volume V of the Fletcher Report.

In particular, SDIO should evaluate the concept of a dedicated communication network. If the evaluation shows promise, a prototype should be designed, implemented, and deployed for final evaluation. The emphasis should be placed on low cost via mass production and deployment techniques.

2. Network Control

The behavior of any large dynamic network can not be fully predicted analytically or by simulations. Therefore, it is necessary to have the ability to monitor and control the entire network's behavior, including performance, connectivity, activity levels, and blockages.

Developing a man-machine interface to visualize the network's dynamic status is necessary to allow human operators to recognize and detect unpredicted, undesired patterns and to initiate corrective actions.

Simulation affords an objective method of evaluating alternatives. This method, which is necessary due to the complexity of the communication system and by the interaction among its components, allows the study of various system and deployment options. A communication system simulator should be designed such that the battle management simulators (see Chapter IV) can incorporate it as a part of total system simulation.

After system validation by simulations, there would be a need for more detailed prototyping. This may be accomplished by a system consisting of a small number of satellites using early models of the final communication system (*i.e.*, hardware, software, and algorithms).

SDIO should develop early such a spaceborne prototype network.

3. Communication Protocols

The importance of a unified set of consistent, compatible protocols can not be overstated. SDIO should take advantage of the substantial body of existing experience in this domain.

The system's ability to deliver information (as opposed to bits and data) depends mainly on the protocols used. The protocol suite should be designed to be open and extensible. It should be compatible with other software interprocess communication (IPC) procedures, simulation conventions, and other internetworking requirements. The earlier a commitment is made to a protocol scheme the easier it will be to achieve the desired interoperability among distributed software, simulations, and internetworking with other computer communication networks.

Protocols are the fundamental means to facilitate IPC. They act as the "language" that processes use to "program" other processes to perform certain tasks. A properly designed set of protocols could handle various IPCs in a consistent manner, independent of the spatial distribution of the processes. They should handle all the communication details, including routing, error handling, and flow control, without requiring user participation (unless so desired).

The protocols to be used over the spaceborne communication network should be developed as extensions of the IPC procedures, with special attention paid to the unique networking requirements of the strategic defense system, such as real-time traffic, type of service, and priorities. These protocols should also be compatible with the general DoD internetworking protocols.

There are several aspects in which the SDI protocols differ from most conventional or traditional protocols, such as those used in other DoD packet switching networks (*e.g.*, ARPAnet and MILnet). The SDI protocols must deal with requirements for both high reliability (such as for weapon release) and also for real-time sensor communication, which has totally different characteristics. The SDI protocols must also be able to cope with significant variance in communication performance, such as those caused by hostilities, because this will occur when high performance is needed most.

The interoperation between the strategic defense network and other DoD networks should be addressed early. Plans for interneting should be studied, especially with other DoD spaceborne networks, for providing mutual backup capabilities.

The adapted communication protocols will affect the architecture, software, and simulation methods.

SDIO should address early the development of protocols oriented toward the special requirements of strategic defense.

4. Technology

The existing communication technology can not support the special requirements of the envisioned strategic defense system.

There is already a dangerous disparity between the data rates available for point-to-point communication and the performance of store-and-forward packet processing nodes. The data rate of existing packet switches and network management algorithms is lagging behind the transmission technology, and must be improved.

Survivable networking of satellites is still an open research issue. The difficulties are due to the dynamic configuration required by the changing topology caused by the difference of velocities and orbits,

Both power and security considerations suggest the use of highly directional, narrow-beam communication. Narrow-beam communication necessitates exact spatial and temporal coordination between all communicating assets, which may not be possible (*e.g.*, because assets can not generally predict when others need to initiate a transmission to them). To overcome this difficulty, an omnidirectional means to initiate communication is preferred, followed by unidirectional transmission (*e.g.*, beam forming and retrodirectional transmission).

The transmission technology requires more research and development in order to:

- Increase the communication and packet processing data rates.
- Explore advanced transmission media (*e.g.*, lasers and integrated microwave devices).
- Improve robustness and resistance to electronic countermeasures (ECM).
- Reduce detection probability.
- Develop adaptive error-correction/data-rate tradeoff.
- Improve security.

5. Security

The security requirements of the strategic defense system are more severe than those of other systems because of its operational characteristics and its unique nature as an unattended system. The security of transmissions (TRANSEC) and communications (COMSEC) for unattended spaceborne systems is much more complex than for traditional systems. Special attention must be paid to issues such as external override and reload capabilities, key distribution, and access control. New approaches to the security of computer communication must be developed and implemented.

The panel concludes that existing security systems (concepts, procedures, and devices) are not suitable for strategic defense and that new systems must be developed and produced.

SDIO must pay attention early to the development and production of the required security systems. This is expected to be a long process even with the help of the National Security Agency.

6. SDInet

The ARPAnet is a successful computer network that supports resource sharing and exchange of information. SDIO should learn from the success of the ARPAnet and establish a computer communication network to facilitate cooperation, information exchange, and resource sharing among the SDI contractors.

This proposed SDInet should support intercomputer communication to facilitate resource sharing, such as simulators, design tools, fabrication of advanced circuits, and access to supercomputers.

This SDInet should also support intercomputer communication at various user-oriented levels, such as support of electronic mail, file transfer, and virtual terminal connections, in addition to traditional lower-level connections (such as at the packet level) that are offered by most networks today. SDIO should provide its contractors with access to SDInet via nodes that process messages and provide user-oriented services.

SDInet will serve as an early testbed for experiments in distributed systems that are basically similar to a distributed strategic defense system, such as the battle management and spaceborne communication systems. SDInet could serve as the proving grounds for the SDI protocol development.

An important payoff of such a network is the formation of a technical community and its resultant cooperation. The benefits are substantial and appear in the form of wide cooperation and interoperability due to sharing of software and other resources made possible by the network. For example, giving contractors immediate access to special expensive resources (such as multiprocessors and supercomputers) through the SDInet will save not only funds, but also many months of project delay.

SDInet will help contractors to cooperate in many ways, formally or informally, such as by testing new software on simulators located at other sites. SDInet should become the primary way for contractors to verify the interoperability of their software and to access various simulators. It will provide access to other networks through its gateways to MILnet and ARPAnet.

SDInet will also alleviate many of the time-consuming management chores such as various demonstrations, coordination, submission of reports, and contract update negotiations.

SDInet should be built to carry non-classified information. However, it should be able also to carry "black" classified information.

B. Recommendations

The panel recommends that SDIO:

- Initiate a study of communications architectures beyond those implied by Volume V of the Fletcher Report. In particular, evaluate the concept of a dedicated communication network. If this evaluation shows promise, a spaceborne prototype network should be designed, implemented, and deployed for final evaluation. The emphasis should be placed on low cost via mass production and deployment options.

- Pursue research and development of the algorithms and protocols for operating the recommended store-and-forward communications network. SDIO should address the development of protocols as early as possible.
- Pursue research and development for advancing the communication technology toward the special requirements of the envisioned strategic defense system. Pay attention early to development of new security systems.
- Develop simulation of the spaceborne communication networks to study system and deployment options.
- Establish a computer communication network (SDInet) to facilitate cooperation and information exchange among SDIO contractors.

C. Summary

Research and development are required for general communications architectures, for spaceborne networking, for SDI-oriented protocols, and for advancing the spaceborne transmission technology. The entire security issue requires special early attention because this is expected to be a *very* long process.

The concept of a dedicated spaceborne communications network should be investigated.

SDIO should install a computer communications network to improve cooperation among its various contractors, and to serve also as a testing grounds for distributed system architecture.

VIII. PROGRAM MANAGEMENT

It is clear, given the scope of the technology areas that require investigation, that the BM/C^3 technology program presents SDIO with a large and complex management problem. Such a program requires technical direction, progress review and evaluation, and effective program management and contracting practices.

A. Discussion

1. Technical Direction

To provide technical direction, SDIO must review research plans and evaluate specific approaches and proposals. This aspect of research program management requires a thorough understanding of the interdependence of research efforts under its direction. This interdependence includes, for example, knowing when one effort needs the results of another and understanding how a delay or advance in any particular research effort affects the remainder of the technology program.

This aspect of research program management also includes the foresight to create the technical infrastructures that would be necessary later for communication and access to advanced fabrication facilities.

SDIO-SY should maintain strength in computer science within its program management staff in order to understand computer science issues as they relate to BM/C^3 and to be able to structure its program to address relevant issues.

In addition, SDIO should retain an ongoing panel of scientific advisors to provide help in specific areas of computer science and to review the BM/C^3 technology program. This advisory panel would provide SDIO with advice concerning its overall technology program, review plans and evaluate approaches and proposals, monitor progress of technology and system development efforts, and provide independent critique. Such a panel would also be able to provide SDIO with a valuable link to technology and research efforts at universities and national laboratories.

The development of the SDInet, as recommended in Chapter VIII (Communication), would be one positive step to encourage the development of a common technical infrastructure between SDI contractors. It would also provide a resource-sharing mechanism and promote the creation of an SDI community for software development.

2. Program Management

SDIO needs to review and evaluate research efforts to maintain program performance and provide program redirection when necessary. Review and evaluation of research papers, technical reports, and test results may provide the information needed to determine how best to proceed with each element of the technology program.

The panel recommends that SDIO use independent contractors to perform the technical functions of program managers in those areas in which SDIO can not enlist personnel of the appropriate technical caliber. This use of contractors would allow SDIO to tap the talent of leading researchers in the scientific community.

These external experts will be able to provide SDIO with technical expertise that would not otherwise be available within its organization.

These experts should be independent of the major contractors to assure that they would be unbiased, with no other objective than to develop the most effective SDI technology and systems within fiscal and political constraints.

In addition, SDIO should routinely assign to each contract another contract to cross-check it and to validate its results and its deliveries.

3. Contracting Practices

Because this is a research and technology development program, one can expect the research emphasis to change throughout the program's duration in response to the results of specific research efforts. It is essential, therefore, that SDIO have the management freedom and flexibility needed to adapt its program in response both to technological evolution and to delays in technology advances. Program management tools and contracting practices must support the provision of this management freedom and flexibility.

The program directed by SDIO must adapt to a rapidly evolving and changing environment. As a result, SDIO needs a program management structure and contracting method that allows it to alter and adapt its program as rapidly as technology issues are resolved. Automated means of tracking program interdependencies are needed. Special contracting methods or clauses or changes in specific DoD program management guidelines are needed. For example:

- Computerized contracting procedures to facilitate the contracting process and reduce procurement delay and paper flow.

- Contracting practices that encourage capitalization for both hardware and software development.

- Contract award criteria that encourage contractors to employ the best quality personnel – cost should not be the prime factor for awarding level-of-effort contracts.

Therefore, SDIO should sponsor research to achieve responsive program management and contracting practices.

B. Recommendations

The panel recommends that SDIO:

- Maintain strength in computer science within SDIO-SY, and retain an independent, ongoing scientific advisory panel.
- Contract independent technical program managers, when needed.
- Use a computer communication network to encourage the development of a technical community of the SDI contractors.
- Sponsor research in program management and contracting.

C. Summary

Research emphasis in the BM/C^3 technology is likely to change in response to evolving technological capabilities. Innovation is required to assist SDIO in managing this complex technology program.

In particular, SDIO should be free to contract independent technical experts to provide the required expertise for program management and contractor performance evaluation that would not otherwise be available within SDIO.

SDIO should sponsor research to develop program management and contracting practices that will allow SDIO to alter and adapt its programs as rapidly as technology issues are resolved.

ANNEX A: MEMBERSHIP: EASTPORT 1985

Study Chairman:	
Dr. Danny Cohen	Information Sciences Institute University of Southern California Marina del Rey, California
Members:	
Dr. Bijoy Chatterjee	Advanced Technology Directorate ESL Sunnyvale, California
Dr. Richard Lau	Scientific Officer Office of Naval Research Arlington, Virginia
Dr. Richard J. Lipton	Professor, Department of Computer Science Princeton University Princeton, New Jersey
William G. MacLaren Maj Gen USAF (Ret)	Vice President, V. Gerber Associates Arlington, Virginia
Dr. David Mizell	Scientific Officer Office of Naval Research Pasadena, California
Dr. Charles L. Seitz	Professor of Computer Science California Institute of Technology Pasadena, California
Dr. Alan J. Smith	Associate Professor, Computer Science Division University of California Berkeley, California
Technical Support:	ANSER Arlington, Virginia
Walter L. Brothers	Senior Weapons System Analyst
Anita L. Crossland	Editor
Dr. William R. Stout	Senior Mathematician

ANNEX B: ACRONYMS LIST

ARPAnet	The communications network of DARPA
BM/C^3	Battle Management/Command, Control, and Communication
BMD	Ballistic Missile Defense
DARPA	Defense Advanced Research Project Agency
DEW	Directed Energy Weapons
DoD	Department of Defense
ECCM	Electronic Counter Countermeasures
ECM	Electronic Countermeasures
EMP	Electromagnetic Pulse
IOC	Initial Operational Capability
IPC	Inter-Process Communication
KEW	Kinetic Energy Weapons
MILnet	An operational military packet switching network
MOR	Military Operational Requirement
MOSIS	DARPA's MOS Implementation Service
MSS	Multi-Satellite System
MTBF	Mean Time Between Failures
NTB	The National Testbed facility of SDIO
ONR	Office of Naval Research
PPBS	Planning, Programming, and Budgeting System

RADC	Rome Air Development Center
RV	Reentry Vehicle
SATKA	Surveillance, Acquisition, Tracking, and Kill Assessment
SDI	Strategic Defense Initiative
SDIO	Strategic Defense Initiative Organization
SLBM	Submarine-Launched Ballistic Missile
VHSIC	DoD's Very High Speed Integrated Circuits program
VLSI	Very Large Scale Integrated circuits